Titre: Title:	Secure Access Control Architectures for Multi-Tenancy Cloud Environments
Auteur: Author:	Hamid Shayegannia
Date:	2015
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Shayegannia, H. (2015). Secure Access Control Architectures for Multi-Tenancy Cloud Environments [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/1671/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/1671/
Directeurs de recherche: Advisors:	Alejandro Quintero, & Ronald Beaubrun
Programme: Program:	Génie informatique

UNIVERSITÉ DE MONTRÉAL

SECURE ACCESS CONTROL ARCHITECTURES FOR MULTI-TENANCY CLOUD ENVIRONMENTS

HAMID SHAYEGANNIA
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
JANVIER 2015

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé:

SECURE ACCESS CONTROL ARCHITECTURES FOR MULTI-TENANCY CLOUD ENVIRONMENTS

présenté par : SHAYEGANNIA Hamid

en vue de l'obtention du diplôme de : Maîtrise ès Sciences Appliquées

a été dûment accepté par le jury d'examen constitué de :

Mme BELLAÏCHE Martine, Ph. D., présidente

M. QUINTERO Alejandro, Doct., membre et directeur de recherche

M. BEAUBRUN Ronald, Ph. D., membre et codirecteur de recherche

M. PIERRE Samuel, Ph. D., membre

DEDICATION

To Sepehr, my beloved son who brings light, hope and color to my life.

To Majid, my brother, for his kindness and devotion, and for his endless support whenever I needed; his selflessness will always be remembered.

To my mom, she taught me all about real sacrifice.

ACKNOWLEDGEMENTS

To my director of research, Prof. Alejandro Quintero, who provided me the means, the supervision and the pieces of advice to successfully complete this project.

Further acknowledgement and thanks is due to my co-director of research, Prof. Ronald Beaubrun at The University of Laval Quebec City. His support was truly helpful and greatly appreciated.

Very special thanks are due to my friends and colleagues Mr. Mohab Aly, Mr. Shahin Vakilinia and Mr. Afshar Ganjali for their help and support.

To all my “Polytechnique” colleagues who were there whenever I needed any help and advice. I really appreciate.

To my second family here in Canada, to “Naghmeh Sabet”, “Khosro Shemirani”, “Fahimeh Sinai”, “Peyman Rajabian Esfahani”, “Hossein Mirzakhani” and “Pegah Mousavi”. They helped me morally.

To my manager in Morgan Stanly Mr.Arturo Reyez who gave me the chance to work and study at the same time.

RÉSUMÉ

L'Infonuagique est un paradigme de système informatique distribué qui offre la possibilité aux usagers (clients) d'accéder à des services et ressources partagés hébergés chez des fournisseurs, afin de mieux répondre à leur besoin en matière de service et d'infrastructure informatiques. Dans l'environnement infonuagique, une même machine ou serveur physique peut héberger plusieurs machines virtuelles (VMs) qui sont partagées entre différents usagers ou clients, rendant ainsi transparent le partage des ressources matériels.

De ce fait, l'Infonuagique crée un environnement propice à des cibles faciles, vulnérables et sujettes à des attaques accrues de pirates informatiques. A cause de la complexité des contrôles d'accès et de la difficulté à surveiller les interconnexions entre les différents systèmes, les applications et les données, l'on s'expose à de nouvelles opportunités. Il ne fait aucun doute que, en termes de sécurité, le plus grand défis auquel les fournisseurs et clients sont confrontés dans l'environnement Infonuagique multi-usager est le contrôle d'accès.

La prévention des accès illicites et non autorisés aux ressources infonuagiques passe par un mécanisme de contrôle efficace des accès. D'un côté, les techniques de contrôle d'accès conçues originalement pour des systèmes locaux d'entreprise ne sont pas appropriées à l'Infonuagique et au système de colocation. D'un autre côté, un mécanisme de contrôle d'accès bien conçu ne devrait pas surcharger le système d'Infonuagique et devrait s'adapter avec facilité à l'infrastructure existante. De nos jours, on se fie au VLAN et Coupe-feu par exemple pour assurer le contrôle d'accès dans l'environnement infonuagique. Ces techniques sont tout à fait efficaces mais des techniques complémentaires spécifiques à l'Infonuagique sont nécessaires pour prévenir les accès non autorisés aux ressources partagées dans le système distribué.

Dans le cadre de ce projet de recherche nous proposons CloudGuard, un système qui implémente un mécanisme de contrôle d'accès basé sur un hyperviseur. Suivant le concept de sécurité en profondeur (security-in-depth), CloudGuard ajoute une couche complémentaire de sécurité aux environnements en colocation de l'infonuagique et prévient les accès non autorisés et illicites aux ressources infonuagiques. Cette architecture de sécurité peut être simplement appliquée à l'hyperviseur et fourni un contrôle d'accès évolutif et plus robuste que les techniques basées sur les réseaux existants.

ABSTRACT

Cloud Computing is a distributed computing paradigm which allows the users to access the services and shared resources hosted by the various service providers, to meet their services or resources requirements. In a multi-tenancy cloud computing environment, multiple virtual machines (VMs) are collocated on the same physical server. In such system, physical resources are transparently shared by the VMs belonging to multiple users.

Cloud computing also creates a suitable environment for easy targets, vulnerable and prone to sophisticated attacks. Also, due to the complexity of access and difficulty in monitoring all interconnection point between systems, applications and data sets, this can create new targets for intrusion. Undoubtedly, one of the most important security mechanisms in multi-tenancy cloud computing environment is access control.

Implementing a proper access control mechanism can lead us to prevent unauthorized or illegal access to cloud resources. In one hand, most of current access control techniques were originally designed for enterprise environments that do not consider the characteristics of cloud computing and multi-tenancy environments. On the other hand, a well-designed access control mechanism should impose less possible overhead to the cloud computing system and it should easily leverage with the existing cloud infrastructure. Today, VLANs and firewalls are example of techniques that provide access control for cloud environments. These techniques are definitely effective but we need complimentary techniques that fit cloud computing and prevent unauthorized access to the resources in the distributed system.

In this research project we propose CloudGuard, a system that implements a hypervisor-based access control mechanism. Based on the concept of security-in-depth, CloudGuard adds another layer of security to multi-tenancy cloud computing environments and prevents unauthorized and illegal access to the cloud resources. This security architecture can be simply implemented to hypervisor and provide scalable and more robust access control than existing network-based techniques.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

ACLs	Access Control Lists
AWS	Amazon Web Service
CC	Cloud Computing
CPU	Central Processing Unit
CSC	Cloud Service Consumer
CSP	Cloud Service Provider
CSU	Cloud Service User
DAC	Discretionary Access Control
DDoS	Distributed Denial of Service
DHCP	Dynamic Host Configuration Protocol
DoS	Denial of Service
IaaS	Infrastructure as a Service
ICMP	Internet Control Message Protocol
IdP	Identity Provider
IDS	Intrusion Detection System
IP	Internet Protocol
IPS	Intrusion Prevention System
IT	Information Technology
KVM	Kernel-base Virtual Machine
LAN	Local Area Network
MAC	Mandatory Access Control
MTACM	Multi-Tenancy based Access Control Model
NIST	National Institute of Standards and Technology

P-Int	Probability of Intrusion
PaaS	Platform as a Service
RB-MTAC	Role Based Multi-Tenancy Access Control
RBAC	Role Based Access Control
RTT	Round Trip Time
SaaS	Software as a Service
SPI	Software, Platform and Infrastructure
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VLAN	Virtual Local Area Network
VM	Virtual Machine
VMM	Virtual Machine Monitor

CHAPTER 1 INTRODUCTION

Nowadays, a number of computing services, such as data storage and processing, email handling and web content management, are available instantly, commitment-free and on demand, using the concept of cloud computing [1]. Cloud computing is a flexible, cost-effective and proven delivery platform for providing business and consumer services over the Internet [2]. Such platform is shared and utilized by multiple customers who share computing resources, including CPU time, network bandwidth, data storage space, with other users, which refers to multi-tenancy [3]. By multi-tenancy, clouds provide simultaneous, secure hosting of services for various customers utilizing the same cloud infrastructure resources [4]. One customer can gain unauthorized access to the information of other customers. In this context, it is important to control the access of network entities to the information. This thesis investigates secure access control mechanisms in multi-tenancy cloud environments. This chapter provides the reader with an overview of the thesis and addresses the basic concepts, the problem definition, the objectives as well as the organization.

1.1 Basic concepts

This section offers an introduction to the basic concepts related to the problem definition and objectives. Firstly, a brief survey of cloud computing is presented. Later on, multi-tenancy in the context of virtualization is introduced as a technology and is explained in further details. Then, a sufficient concern will be given for the main aspects of security. Finally, access control will be explained.

1.1.1 Cloud computing

Since a huge amount of services is available online, the use of distributed systems namely cloud computing, is growing and is becoming more and more popular. Cloud computing is defined as a large scale distributed computing paradigm which has five key characteristics [5] [6]:

- On-demand self-service: cloud is a large-scale pool of resources. Users need to buy the services as they demand.
- Ubiquitous network access: a cloud is a group of virtualized devices, which enables users

to acquire the application service from any location.

- Cost effectiveness: by switching from traditional network to cloud networks, companies can reduce their costs. Because they do not need to buy the whole infrastructure, instead they pay per use.
- Scalability: The size of cloud network can be scaled in a matter of few clicks, because users can add computing computer resources and scale their network easily just by demanding the cloud provider.
- High commonality: cloud computing is not aimed at specific applications. Different applications can utilize the same cloud resources.

To analyze and describe cloud-based systems, many people refer to cloud solutions in terms of its deployment model and services model. These two terms originated from the National Institute of Standards and Technology (NIST) [7] [8]. Cloud infrastructure may be operated in one of the following deployment models: public cloud, private cloud, community cloud and hybrid cloud.

- Public cloud: the computing resources are made available to the general public over a public network. In this context public cloud may be owned, managed, and operated by a business, academic, or government organization or some combination of them. With public cloud services, users do not need to purchase hardware, software or supporting infrastructure, which is owned and managed by providers [9]. (Figure 1-1)

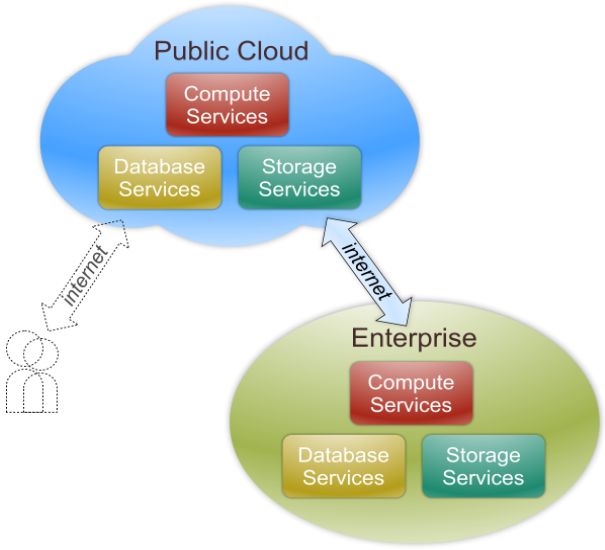

Figure 1-1 Public cloud

- Private cloud: the exclusive access to and usage of the infrastructure and computational resources is given to a single cloud consumer's organization. In this context a private cloud may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises [8]. Private clouds have some key characteristics such as: highly automated management of resource pools, sophisticated security and self-service interface that helps IT staff to allocate IT resources very quickly [9]. (Figure 1-2)

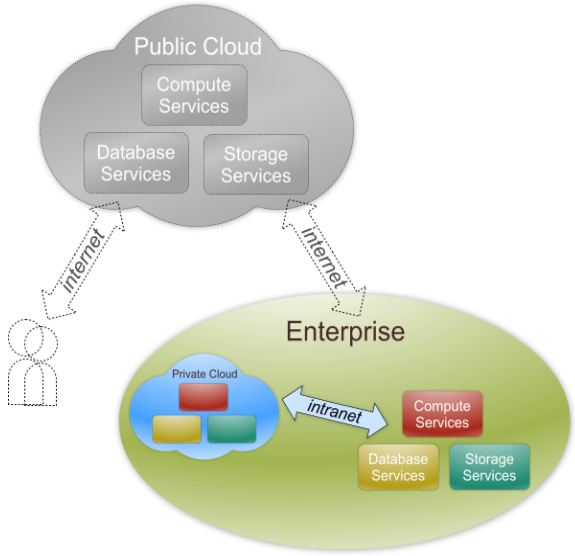

Figure 1-2 Private cloud

- Community cloud: serves a group of cloud consumers which have shared concerns such as mission objectives, security, privacy and compliance policy, rather than serving a single organization, as does a private cloud. It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises. (Figure 1-3)

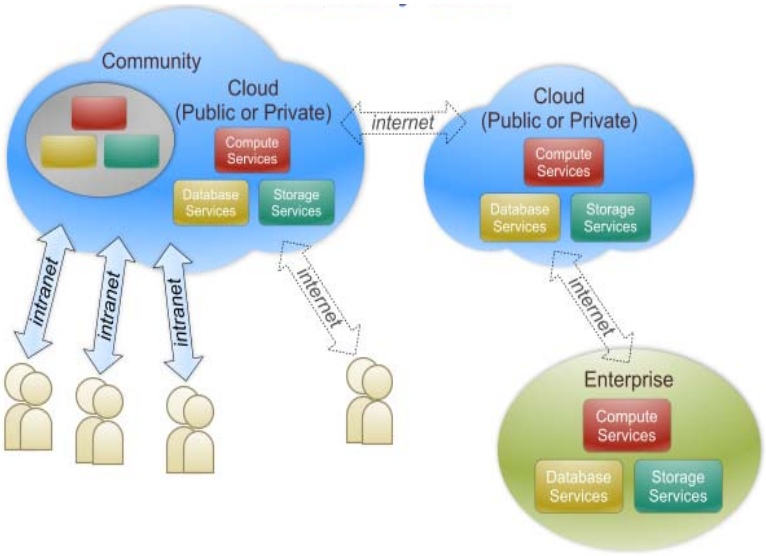

Figure 1-3 Community cloud

- Hybrid cloud: is a combination of two or more clouds (private, community or public) that remain as distinct entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds. (Figure 1-4)

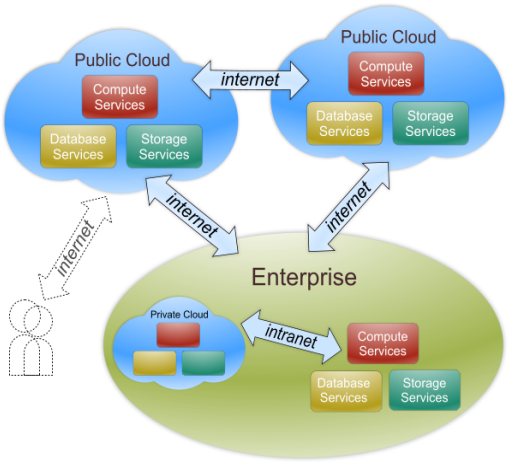

Figure 1-4 Hybrid cloud

Moreover cloud computing provides many service resources over the Internet. These resource services can be readily broken down into one of three **Service Models** as defined by NIST [7] [8] and known as the SPI (Software, Platform and Infrastructure) model. (Figure 1-5)

Software as a Service (SaaS): In this service model, the consumer uses the provided applications and do not manage or control the network, servers, storage and the applications. Examples of SaaS are: Flickr, Google Docs, Siri, Amazon and Cloud Drive [10].

Platform as a Service (PaaS): This solution provides a collection of hardware and software resources that developers use to build and deploy cloud-based applications. PaaS solutions run a Windows- or a Linux-based operating system and normally support a specific programming environment, such as .NET or Java. [11]. The examples of PaaS are: Google App Engine, Amazon Web Services.

Infrastructure as a Service (IaaS): In the infrastructure as a service, the consumer gets access to the infrastructure in order to deploy their applications and systems, but they do not manage or control the infrastructure and they control the storage and applications. Examples of IaaS are: Amazon Elastic Compute Cloud (Amazon EC2) [10].

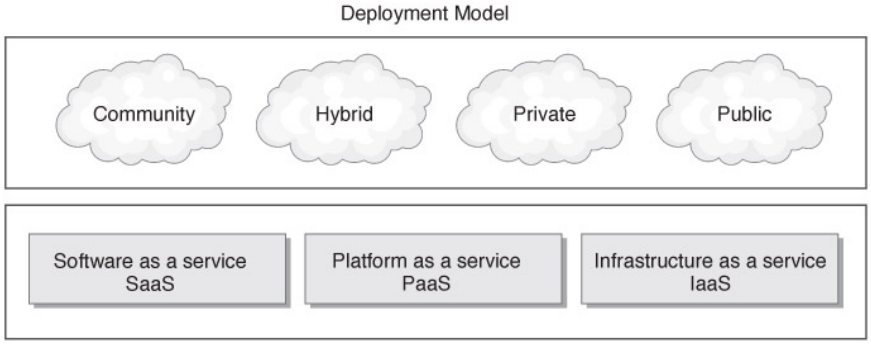

Figure 1-5 Cloud deployment model [9]

1.1.2 Virtualization

Virtualization is an indispensible part of cloud computing. It brings scalability to the cloud computing. By virtualization, the sufficient amount of computing power allocates to the client needs. But if the needs grow, more computing power can be allocated to that client [12]. Virtualization is an extremely powerful tool to grow the scalability. In other words, virtualization consists of creating a virtual version of resources, such as a server, storage device, network or

even an operating system. Li Yunfa et al. (2010) [13] states that " by using the virtual technology, the computer system can aggregate all kinds of data resources, software resources and hardware resources and make there resources to provide service for different tasks". Virtualization technology acts as the core concept of cloud computing. Thus, we should pay special attention in the architecture and security of virtualization when researching about cloud computing security.

Hypervisor also called Virtual Machine Monitor (VMM) is one of the key components of virtualization and support the running of multiple Operating Systems (OSs) concurrently on a single host computer. The main responsibility of Hypervisor is to managing the application's OSs that is called the guest OSs and their use of the system resources (e.g., CPU, memory and storage). Figure 1-6 shows the architecture of a virtual machine system in which multiple virtual machines (VMs) share the same "physical machine", or host. Above the hardware layer, the hypervisor provides resource allocation to virtual machines [13].

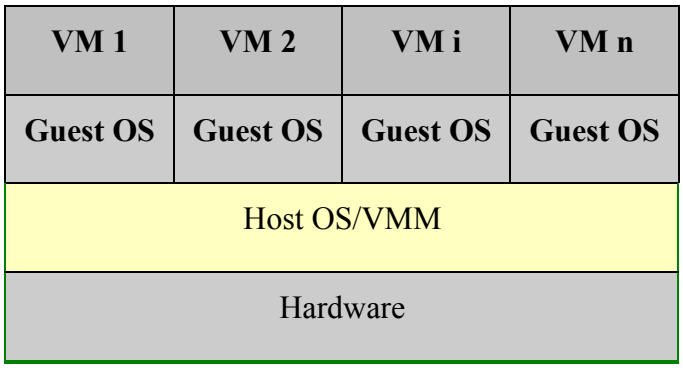

Figure 1-6 Architecture of virtual machine system [13]

Multi-tenancy in virtual machines is another important attribute of cloud computing. Multi-tenancy is a technology by which several VMs are runnable in a physical server. In a multi-tenancy environment, multiple customers share the same application, running on the same operating system, on the same hardware, with the same data-storage mechanism. In other words, different users are assigned different VMs that are running on a same physical hardware. Multi-tenancy has several advantages such as effectiveness in the resource sharing and also prioritizing users based on their needs. Multi-tenancy in VMs can be considered as a hierarchical model, where appropriate policies are enforced on the VMs at every level leading to better governance and segmentation of the consumers [14].

1.1.3 Security

Security in cloud computing as defined by [10] refers to a broad set of policies, technologies, and controls deployed to protect data, applications, and the associated infrastructure of cloud computing. More specifically, cloud computing security refers to three aspects as illustrated in Figure 1-7: confidentiality, integrity and availability [15].

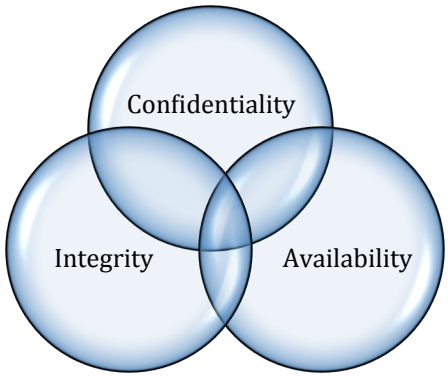

Figure 1-7 Aspects of security

Confidentiality refers to the fact that only authorized parties have access to computer-related assets. Having access here means reading, printing, or even knowing that a particular asset exists [15]. Confidentiality becomes very important if the cloud system is dealing with sensitive data. Therefore, maintaining maximum level of data confidentiality in the cloud is a fundamental requirement, which will attract even more users consequently. Basically, for dealing with this challenge in the cloud systems there are two traditional approaches: physical isolation and cryptography [3].

Integrity in a cloud means that data should not be lost or modified by unauthorized users[3]. In the context of data security, it means that only authorized parties should be able to modify the assets. Here, modification means writing, changing, deleting and creating [15]. In most of the cases, the confidentiality and integrity are used interchangeably, because if the confidentiality of data is guaranteed in a cloud system, we can be sure that the integrity is also maintained.

Availability in the context of cloud computing means that assets are accessible to authorized parties at the appropriate time. Nowadays, we can see many cloud environments in

which availability is very important. For example, we can name YouTube and Netflix, which offer multimedia to their customers and for which the availability of information is essential [15].

1.1.4 Access control

Access control is a security feature that controls how users and systems communicate and interact with other systems and resources [16]. Information and resources should be accessed in a way so they can be protected from unauthorized modification or disclosure. Generally, there are three types of controls that enforce access control [16]:

- Physical controls

  Implementation of security measures in a defined structure in order to prevent unauthorized access to sensitive material. Examples of physical controls are: security guards, picture IDs, locked and dead-bolted steel doors, biometrics, closed-circuit surveillance cameras and motion or thermal alarm systems.

- Technical controls

  Technical controls employ technology as a basis for controlling the access to sensitive data throughout a physical structure and over a network. Examples of technical controls are: encryption, smart cards, network authentication, access control lists (ACLs) and file integrity auditing software.

- Administrative controls

  Administrative controls define the human factors of security. All levels of the personnel within an organization are involved in administrative controls. Administrative controls also determine which users have access to what resources and information by the use of: training and awareness, disaster preparedness and recovery plans, personnel recruitment and separation strategies and personnel registration and accounting.

The above-mentioned control types can be integrated into cloud computing security architectures in order to preserve the integrity, confidentiality and availability of resources that are collocated in multi-tenancy cloud environment. In this research we investigate the use of technical controls for controlling the access to sensitive information in a multi-tenancy cloud.

1.2 Problem definition

In a multi-tenancy cloud network, many clients may collocate in one or more hosts. In this context, each client may have one or more VMs. Thus, one physical server can host few VMs. In such an environment, one client can send unlimited amount of traffic to another client. Accordingly, a malicious agent can rent a VM on the same host where the target VM resides. This malicious agent can send unauthorized traffic to target VM and violate the security of the target VM.

The unauthorized traffic may contain some script or malware which violate the confidentiality or integrity of the target VM's data. Even if the traffic does not contain any script or malware, by keep sending numerous amount of simple traffic (e.g., Internet Control Message Protocol ICMP packets) a malicious agent can perform ping-to-death attack. This attack can violate the availability of data.

Sending unauthorized traffic to another VM makes it possible to perform other sort of attack. For instance, the malicious agent that own a VM can perform VM Hopping over another user that is co-located on the same host [17, 18]. With VM hopping, an attacker has control of one virtual machine and tries to gain control of another. VM hopping allows attacker to move from one virtual server to the next or even gain root access to physical hardware [19]. VM hopping is a considerable threat because several VMs can run on the same host making them all targets for the attacker. By performing this attack a malicious user can violate the security and steal the data of the other users that are located on the same server by compromising hypervisor file system [14].

Beside that, the malicious insider can perform DoS (Denial of Service). This attack exhausts the resources of the cloud network such as bandwidth and computing power by sending large amount of unauthorized traffic to another VMs.

1.3 Objectives

Our main objective in this research is to design secure access control architecture for multi-tenancy cloud environments. In order to reach the main objective, we pursuit the following goals:

1. To study and to analyze existing methods of controlling access in the multi-tenancy cloud environments.

2. To propose a secure access control architecture to prevent malicious insiders from generating and sending unauthorized traffic to another VMs in the multi-tenancy cloud network.
3. To evaluate the performance of the proposed architecture and prove that it can overcome the problem using mathematical modeling.

1.4 Organization of the thesis

This dissertation is organized into five chapters. The rest of this dissertation is organized as following: Chapter II presents the state of the art of the access control models in the virtualized environments and, more especially, in the multi-tenancy cloud networks. Chapters three and four present information relative to experiments conducted for this project. In chapter III the proposed architecture is exposed in details. Chapter IV is devoted to the validation and performance analysis of the proposed architecture. Finally, we conduct the conclusion and the future works.

CHAPTER 2 STATE OF THE ART

This chapter provides the state of the art of existing security architectures for controlling access in multi-tenancy cloud environments. First of all we describe multi-tenancy in detail. Then we will explain the problems and issues in the multi-tenancy environments. Third section is dedicated to security requirements. After that we study the access control methods in the cloud computing. Finally, problems and challenges of the existing multi-tenancy cloud security solutions will be discussed.

2.1 Multi-tenancy

Multi-tenancy is the core technology of cloud computing [20] . It means the use of the same resources by multiple consumers; these consumers may belong to the same organization or to different organizations. The best way to think about multi-tenancy environment is to see each client or user of cloud as the tenant of that cloud. Both cloud service providers (CSP) and legislative bodies need to acknowledge the impacts that multi-tenancy can have on user privacy.

Guo et al. (2007) believe that multi-tenancy technology is one of the key competencies for network delivery services to achieve higher profit margin by leveraging the economics of scale [4]. This model, enables many users, data and resources to be located in the same computing cloud, and these data and resources are controlled and distinguished through the use of tagging for the unique identification of resource owned by individual users [20]. Multi-tenancy in cloud service models implies a need for the following concepts [21]:

- Policy-driven enforcement is an approach for enforcing the security policies. In this approach, the security policies and access controls are implemented in a way that can be managed and handled effectively.
- Segmentation is used to ensure sensitive back-end services are well protected from the (potentially vulnerable) publically available front-end. This segmentation can be accomplished by implementing a hypervisor firewall.
- Isolation means that, virtual machines that belong to a tenant should be isolated from other tenants. This can be achieved by hosting virtual machines of tenant in different physical servers or by using VLAN.

- Governance enables cloud users to customize and tailor software and hardware according to their needs to fit with their specific needs. User in a multi-tenancy cloud environment should be able to easily manage and configure their virtual machines.
- Service levels measure the quality of service that a service provider is supposed to provide to clients. In the context of multi-tenancy cloud computing, the service provider agrees to provide certain type of computing services according to the service levels and contact that was signed between them and cloud users.
- Charge-back/billing models define different billing models for different consumer constituencies. Pay-per-use is one of the characteristics of the cloud computing and by that, cloud users can reduce their costs. Also for cloud providers, in order to make full use of economic scale of cloud computing, the services are usually hosted following a multi-tenancy model [22].

As illustrated in Figure 2-1, a multi-tenant cloud service provider has three essential elements: the cloud manager, the hypervisor and the virtual machines [21]. Cloud manager is a console of management provided for clients in order to manage their cloud infrastructure. Here managing means creating, shutting down, or starting the instances. The hypervisor, also called virtual machine manager (VMM), allows multiple operating systems (guests or virtual machines) to run concurrently on a host server. Its role is to control the host processor and resources, and also to allocate what is needed to each operating system and. Another role of hypervisor is to prevent VMs from using resourcing more than the amount that was allocated to them. A virtual machine is an isolated guest operating system installation within a normal host operating system.

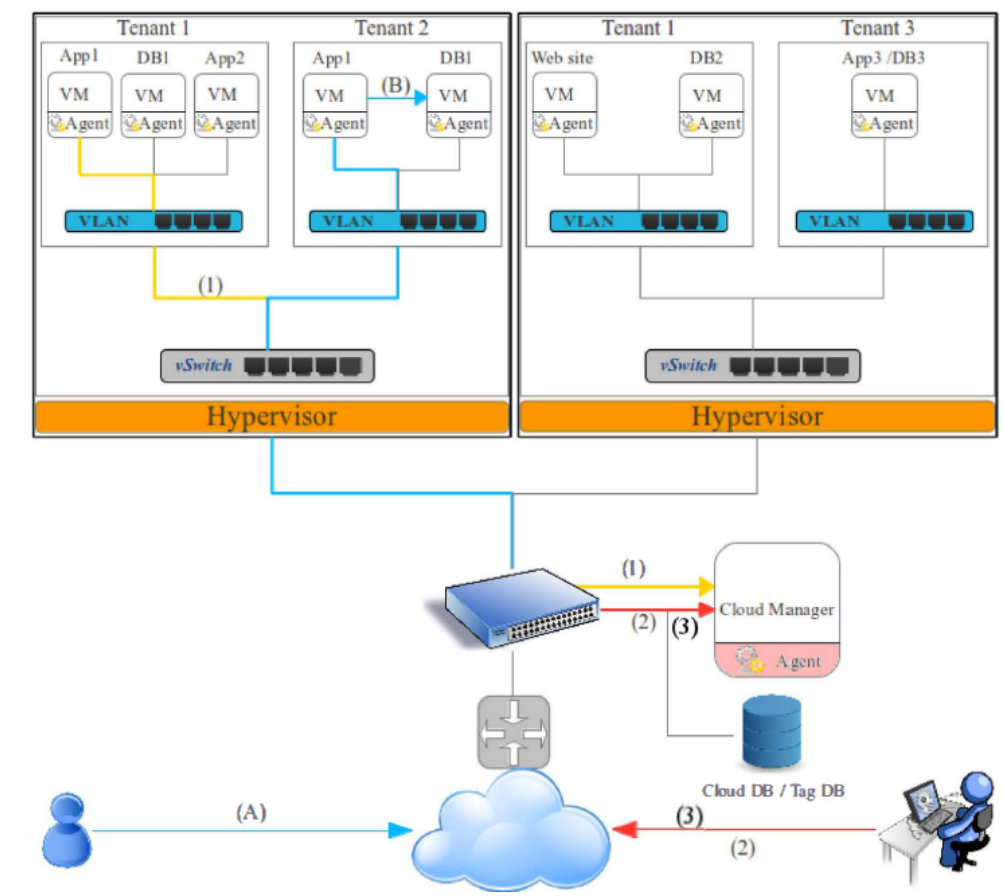

Figure 2-1 A simple model for a multi-tenant cloud service provider [21]

Multi-tenancy architecture for SaaS

To understand multi-tenancy in SaaS cloud computing service model, let us focus on the high level multi-tenancy architecture proposed by Chang et al. [23]. Such architecture has three attributes:

- Scalable

  In this context scalability means the ability to maximize the concurrency of the resources, as well as the ability to use applications more efficiently.

- Multi-tenant-efficient

  Multi-tenancy should have the ability to maximize resource sharing across tenants and to be able to differentiate data belonging to different tenants.

- Configurable

  The ability of configuring applications easily by tenants, without incurring extra development or operation costs for each configuration.

Broadly speaking, multi-tenancy in SaaS cloud application maturity can be expressed using a model with four distinct levels [Figure 2-2]. The term maturity in this context relates to the degree of formality and optimization of processes, from ad hoc practices, to formally defined steps, to managed result metrics, to active optimization of the processes. Each level is distinguished from the previous one by the addition of one of the three attributes listed above.

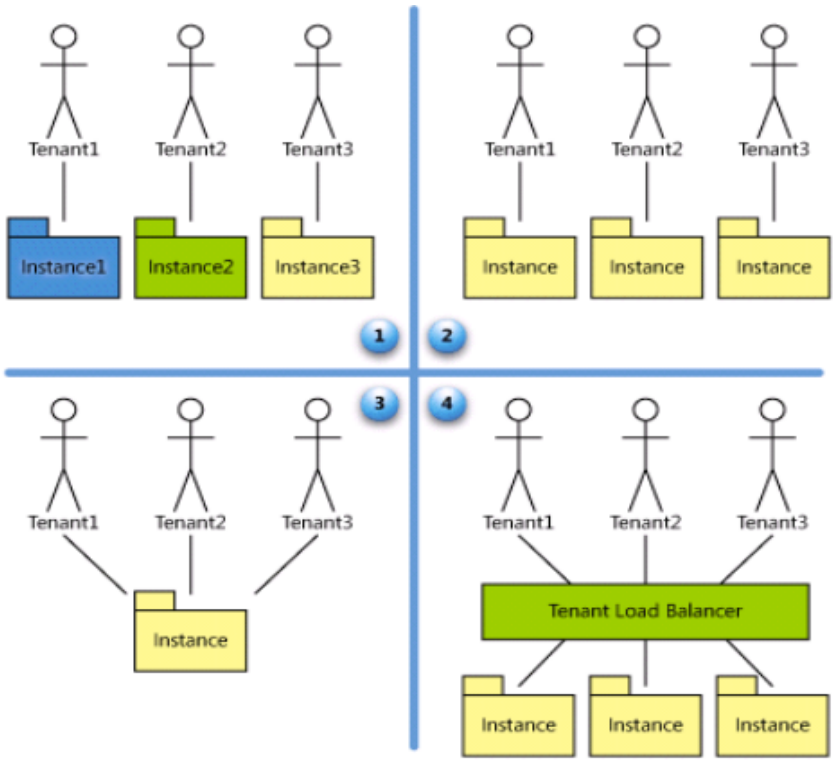

Figure 2-2 Four level SaaS multi-tenancy cloud [23]

- Level I: ad hoc/custom

This level of maturity is similar to the traditional application service provider model of software deliver in 1990s. At this level, each customer has its own customized version of the hosted applications. As well as running its own instance of the application on the host's servers.

- Level II: configurable

At this level of maturity, the vendor hosts an isolate instance of the application for each tenant. Here the cloud provider should provide sufficient hardware and storage to support a potentially large number of application instances running concurrently.

- Level III: configurable, multi-tenant-efficient

At this level of maturity, the vendor runs a single instance that serves every customer. Authorization and security policies ensure that each customer's data is kept separate from that of other customers; and, from the end user's perspective, there is no indication that the application instance is being shared among multiple tenants. Main disadvantage of this level is that it is not scalable. But this approach eliminates the need to provide server space for each instance of each tenant.

- Level IV: scalable, configurable, and multi-tenant-efficient

At this level of maturity, the vendor hosts multiple tenants on a load-balanced farm of identical instances, with each tenant's data kept isolated. A SaaS system is scalable to an arbitrarily large number of customers, because the number of servers and instances on the back end can be increased or decreased as necessary to match demand, without requiring additional re-architecting of the application, and changes or fixes can be rolled out to thousands of tenants as easily as a single tenant.

Choosing a maturity level of multi-tenancy for applications depends on many items, such as business model, architectural model, operational needs and customer considerations [23].

2.2 Security issues in multi-tenancy environments

In a multi-tenancy cloud network, if a security breach occurs, it can result in the exposure of data to other possibly competitive tenants [24]. According to this fact, more attention should be paid to the security of tenant in this cloud environment. For example, if we agree that scalability is one of the key features of cloud computing therefore, matters like scalability and security must be taken into consideration before designing and deploying a multi-tenant environment. In multi-tenancy environments, even the impact of hardware and software failure will also have a much larger impact than that of occurring on a single-tenancy environment [20]. An important impact of multi-tenancy is that the data or trace of operation of a tenant may be

visible (availability) and accessible (integrity and confidentiality) by another tenants [21]. In this context, when the data is visible and accessible by other tenants, it means that the confidentiality and availability of the data is not preserved.

Chow et al. [25] categorized security concerns of the multi-tenancy cloud computing as follows:

- Traditional security concerns means that the network attacks can be committed easier on a cloud network than a traditional network. Among this category we can find different classes of attacks including VM-level attacks. In other words, VM-level attacks exploit potential vulnerabilities in the hypervisor. VM technologies used by cloud vendors are a potential problem in multi-tenant architectures. Vulnerabilities have appeared in VMware, Xen, and Microsoft's Virtual PC and Virtual Server [26].
- Availability concerns prevent companies from switching their traditional networks to the cloud. The stack holders want their network and services to be available all the time. In a virtual environment the same concern exists and the tenants want the services and resources to be available for them in proper time. These concerns center on critical applications and data being available.
- Third-party data control concerns. The legal implications of data and applications being held by a third party are complex and not well understood. In fact, there are some audibility concerns which means, auditing the activities and interaction of the tenants. It seems that there is not sufficient transparency in the operations of the cloud provider for auditing purposes. Currently, this transparency is provided by documentation and manual audits. In march 2009, the information security magazine asks a question that attracts many attentions [27]: "How do you perform an on-site audit for example, when you have a distributed and dynamic multi-tenant computing environment spread all over the globe? It may be very difficult to satisfy auditors that your data is properly isolated and cannot be viewed by other customers."

In addition to previous concerns, Tianfield et al [28] mentioned reusability as an important concern in cloud computing environments. So, reusable infrastructures must be carefully investigated and controlled because they can create a serious vulnerability. Confidentiality and integrity of information could be breached unintentionally, due to data remanence. Data

remanence is the process of residual representation of data that have been in some way nominally deleted or removed. Data remanence could cause unwilling disclosure of private data due to virtual separation of logical drives and lack of hardware separation between tenants on a single infrastructure. Also, malicious agents may claim a large amount of disk space and perform scavenging for sensitive data [28].

2.3 Security methods in cloud environments

In previous section we reviewed security issues in multi-tenancy cloud environments. In this section we are going to study general methods of security in the cloud computing. The reason for dedicating one section for security methods in general in the context of cloud computing is that, is most of the cases we have overlap in different areas of security in cloud.

Privacy of Information in multi-tenancy

K.Wood and M.Anderson (2011) proposed two main solutions for solving the privacy (confidentiality) of information in the multi-tenancy environments [20]. First of all they consider lack of standardized regulations and legislation as a major problem in the field of cloud computing especially in the adoption of multi-tenant systems. The existing European Union (EU) and United States (US) legislations are more specific sector (such as HIPAA for health agencies in the US and SOX Act for financial institutions) on privacy being enforced rather that one consistent privacy agreement being made [29]. Secondly, they name Encryption as a potential solution for confidentiality of information in multi-tenancy environments. Whilst cryptography techniques are appropriate for conserving and protecting users data inside a cloud, there are two drawbacks related to the use of encryption. The first issue arises when the data of all users that are stored in a cloud are encrypted with the same algorithm and this can diminish the confidence of end user about the CSP. The second issue is the fact that basically, the data should be decrypted to plaintext in order to be readable and the process of data decryption can open new doors for malicious agents to intercept processing jobs to access the data. In recent years to new encryption techniques called Predicate Encryption and Homomorphic Encryption have emerged as possible solutions to reduce some of the so-called challenges and concerns of using multi-tenancy environments. Predicate encryption as a new encryption paradigm that provides the master secret key owner with a fine-grained control over access to encrypted data [30].

Homomorphic encryption, on the other hand, has a mechanism that enables us to process the cipher text, in this case we do not need to decrypt data prior to processing [29].

Multi-Tenancy Authorization System

Jose M. Alcaraz et al. (2010) tried to improve the security of multi-tenancy cloud systems by defining a new Authorization model. So, they proposed a multi-tenancy authorization model that suite middleware service in the PaaS layer [31]. This authorization system provides access control to the information and services of all the different cloud services using the cloud infrastructure. When authorization model is created in order to control the access to resources in a cloud system, this authorization model can interpret 5-tuple (Issuer, Subject, Privilege, Interface, Object) in the following way: the Issuer says that the Subject has the Privilege to perform a give action the Object associated to the Interface type, for example (Jose, role(Nige,Admin), Read, CloudStorage, \root\).

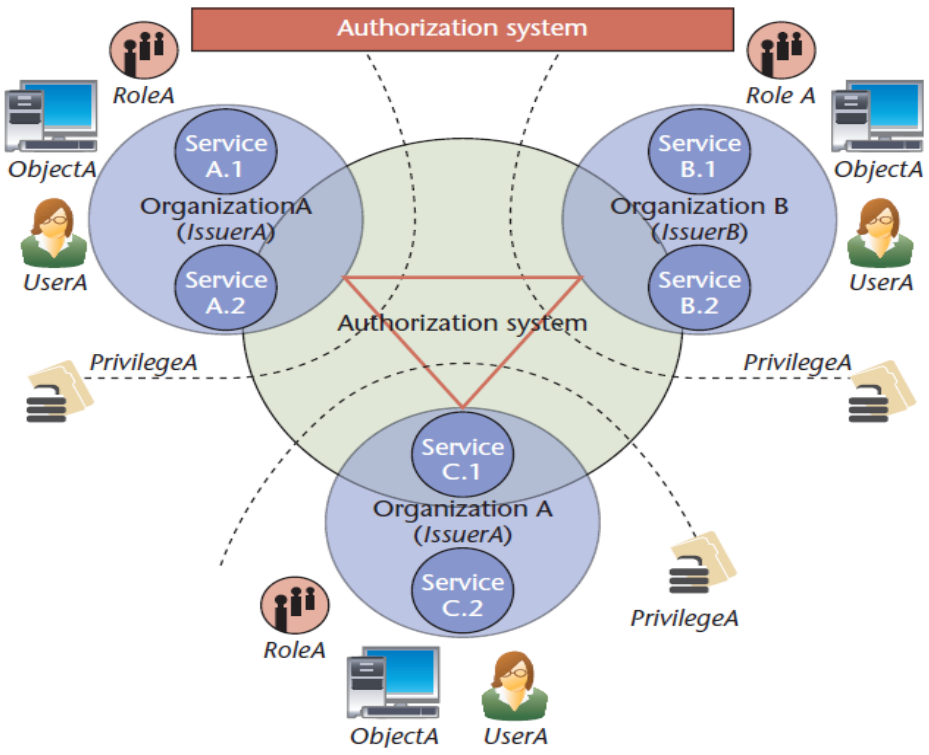

Figure 2-3 Example of an authorization system in a multi-tenancy scenario [31].

Figure 2-3 shows a cloud computing scenario with multi-tenancy authorization system in which, three different businesses exist and each one has two different cloud services. In this

example, the authorization system can interpret the 5-tuple (IssuerA, Role(IssuerB, rusers), Read, ServiceA.1, \root\). In this case IssuerA grants anybody with role(IssuerB, users) Read access to the \root\ folder of the file system provided by ServiceA.1. Also role(IssuerB, users) is controlled by IssuerB. According to this model, different issuers use this model to define the authorization information in the system. When there is an authorization request, the system uses all the authorization information to determine if a request is authorized. If it cannot prove the authorization by using the tuple-5, then a "deny by default' is applied, rejecting access to the resource.

Currently, tenant isolation is a way of securing multi-tenancy environments [32, 33]. Although this isolation should be guaranteed but within a multi-tenant framework, there exist always a risk that the isolation of the tenant is compromised due to a malfunctioning component or architectural weakness or an inadvertent programming error. When an organization is registered as a tenant of a cloud system, it should be able to manage/administer its own users, data and services, but this organization should not be able to administrate functionality of the overall application such as shutdown and viewing system logs [34].

Virtualization security

Dimitri Mckay a security architect at LogLogic Inc., in a published video [33] describes vulnerabilities and security methods in virtualized network environments. As per him, virtualization is the backbone of cloud computing and this virtualization is performed by hypervisor and he describes security challenges in the virtualized networks as below:

- Hyperjacking
- VM-hopping
- VM-theft

Currently, the techniques that exist in order to secure these environments are all traditional security. Basically all these vulnerabilities are coming from hypervisors. In hyperjacking for example when the hypervisor is owned, then the attacker has the ability to steal VMs themselves, the ability to use them as a staging ground to attack other virtual machines and those quest operation systems. Also they can use it as a way of maintaining the persistence. So, when hacker gets the access, they want persistence, they want the ability to come back at any time.

In two other challenges like VM-Hopping and VM-Theft, the flat network plays a vital role. If you think of the way that virtual environments are now flat networks in which they lack intrusion detection system (IDS) and intrusion prevention system (IPS) without routing data through these tools, without log management tools, and actually the networks has become very flat because of these virtualized networks. The networking components that exist in the traditional networks do not exist in the current cloud infrastructures. So, although we think cloud computing is being a new thing, the same concept exists between what was traditional security and what is security. If we consider that cloud is a simple network, one machine can be staging ground to attack high value targets. The challenge here is that if the hacker can compromise a low priority target like an ftp server and using that as a staging ground to attack where the real data lives.

At the end, he focus on the traditional security and propose some solutions for the so called challenges such as, segregating the networks, running the traffic through the IPS and IDS, segregating the virtual networks into different pieces, making sure the traditional routes that people do for traditional networks still applies, making sure that the operating systems are patched, making sure the software solutions in customer side are patched, and most important point is to secure the hypervisor like hypervisor console.

Virtual machine migration

Brian Hay et al. (2012) addressed VM Migration as a point of vulnerability in IaaS model of multi-tenancy cloud environment [35]. The ability to move VMs between physical hosts is vital in order to perform load balancing and also facilitating high availability operations. Other considerations that make such migration appealing are as bellow:

- Ability to use physical hardware efficiently to meet consumer demand. For example, consider the simple case of a CP that offers consumer the option to deploy small (25% of a physical server) or large (50% of a physical server) VMs. If they have three physical servers with 3 small VMs each, they cannot deploy a large VM despite there being 75% of a physical server available in the resource pool. Migration of small VMs would allow the same physical resources to be meet the demand.
- Ability to migrate VMs to alternative geographical locations. This migration prepares VMs for recovery, also to provide better network performance as user demographics

change, or even to place the VM in a legal jurisdiction more compatible with the activities being performed.

- Ability to organize and segregate VMs in order to prevent conflicts, such as when VMs assigned to tow competing corporations are assigned to the same physical host.

Brian Hay et al. in their research, they explore the analogy between human migration in the real world and virtual machine migration in an IaaS cloud environment. So, based on an examination of these analogous real-world scenarios and their applicability to the cloud, they addressed following issues and solutions:

1. Jurisdictional control

   To overcome the jurisdictional control issue, Brian Hay proposed restricting the migration of VMs to those zones that meet the requirements, and vice versa, a more fluid transfer of VMs through the cloud can be achieved while complying with jurisdictional regulations.

2. Zone evolution

   In order to meet the ever-changing needs of the cloud, new servers and resources may be added to cloud zones. Knowledge of characteristics of and requirements of the VMs in the cloud help us for better utilization of cloud resources given the dynamic environment also it can help us to identify whenever a VM in a zone would be better suited elsewhere or when the VM is no longer in compliance with its current zone.

3. Migration

   The process of VM migration like human migration may have different motivational factors. In this migration process, at least two zones are involved, source zone A and destination zone B. Stakeholders of the zones, just like border patrol have the dual goals of ensuring that both entering and departing VMs comply with restriction on their movement. These Emigration Agent guards are responsible for determining if an exiting VM (in case of having SECRET data) meets the requirements for leaving a zone. Just like an immigration perspective, immigration agents (like Border Patrol) should be able to ensure that VMs moving into a new zone meet the migration requirements of that zone.

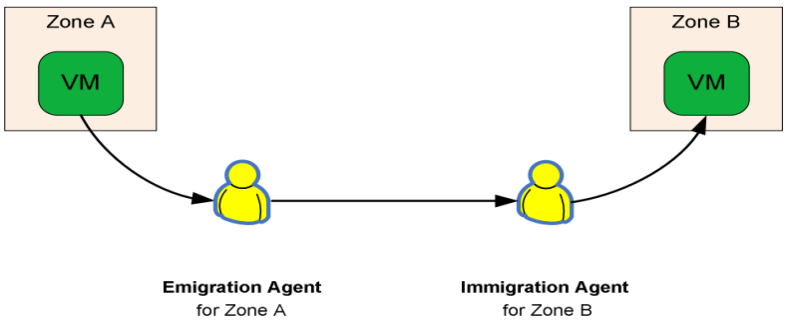

Figure 2-4 Migration path between two zones [35]

4. Resource Instantiation

   Instantiation of a new resource is like migration of VM and all approaches in the migration can be applied to instantiation. So, a new VM can be considered as a special migration case that is coming from a "null zone".

2.4 Security requirements

I.Iankoulova et al. [36] used a security model to analyze and classify the security requirements in cloud computing. This security model consists of nine sub-factors that identify different aspects of system security. In this section we mention six important security requirements that match our research as bellow:

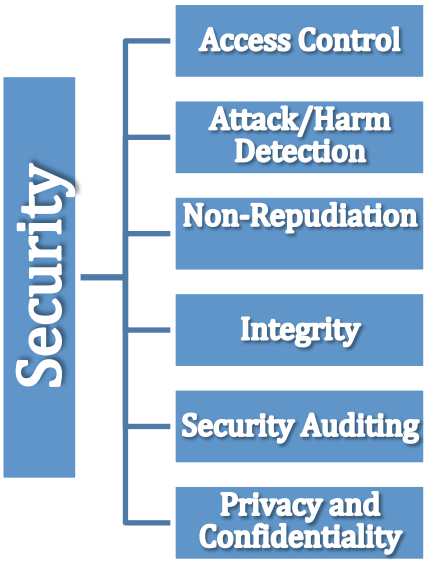

Figure 2-5 Taxonomy of cloud security requirements

I. Access control

In this degree, the system limits the access to tis resources only to authorized parties (e.g., human users, programs, processes and devices). So, this security requirement, address the need to recognize entities that want to interact with the system. Access control also needs to make sure that the parties exactly who they say they are and giving them access only to the resources that they are allowed to access. In order to set the access control requirements, the following items should be taken into consideration:

1. Method of access to the cloud

   Web applications are usually used in order to provide access to the cloud service user (CSU). Unfortunately, due to authentication protocols problems, current web applications are deemed the weakest point of cloud computing [37].

2. Architecture of the cloud

   Virtual machines instance interconnectivity is the biggest challenge in the architecture of the cloud computing. Also a key concern in virtualization is isolation [38], this isolation guarantees that one tenant cannot affect another tenant what is running in the same host.

3. Features of the multi-tenant environment

   In a multi-tenant environment some problems can be cause such as role name conflicts, cross-level management and composition of tenants' access control. So, the security requirements should be aligned to specific context of the multi-tenancy in order to avoid the so-called problems.

II. Attack/harm detection

This security requirement refers to the detection, recording, and notification requirements when an attack is attempted and/or succeeds.

III. Integrity

Integrity is the fact that how well various components of cloud environment are protected from intentional and unauthorized corruption.

IV. Security auditing

For this security requirement, security personnel should be enabled to audit the status and use of security mechanisms by analyzing security related events [39]. Security auditing is usually done by in order to achieve compliance to laws and regulations or also for the sake of accountability and control.

V. Privacy and confidentiality

Privacy as explain before in the first chapter is all about preventing unauthorized parties from obtaining sensitive data and information.

VI. Non-repudiation

Non-repudiation is about preventing a party to an interaction with the cloud to deny the interaction.

Observation to the above security requirements in the cloud computing environments shows that most of the times, some requirements are tightly interconnected. For example, access control and non-repudiation are closely interconnected. Also attack detection requirements are strongly connected to integrity because once being able to identify an attack the step is to stop it from harming the integrity and sometimes the two steps are not separated.

2.5 Cloud access control methods

In cloud computing environments, various entities may appeal to join the cloud and a cloud system consists of large number of entities, such as user and resources. Access Controls are security features that can control how users and systems communicate and interact with other systems and resources [16]. These security features can be implemented into a cloud system in the form of technical, physical or administrative. Access is the gateway that leads to critical assets, due to this fact, access is the most exploited aspect of the security [16]. In order to be more effective, access control should be applied in a layered defense-in-depth method.

2.5.1 Principals

In the field of information security, access is the flow of information between a subject and an object. As shown in Figure 2-6, a **subject** is an active entity (a user, program or process) that requests access to an object or the data within an object. On the other side, an **object** is a passive entity (computer, database, file, computer program, directory or field contained in a table within a database) that contains information or needed functionality [16].

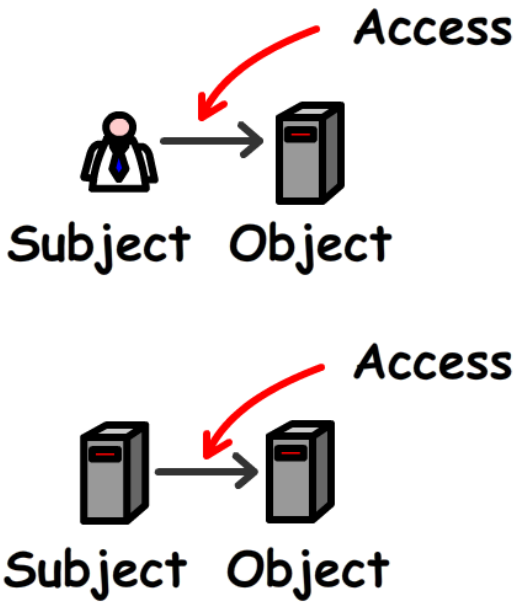

Figure 2-6 Access control principals

Discretionary access control

In a discretionary access control (DAC) model, the owner of the object controls the initial assignment and subsequent propagation of all privileges associated with an object. It means that the data owners decide who has access to resources, and ACLs are used to enforce these access decisions [16]. In a discretionary access control model, access is restricted based on the authorization granted to the users so the users are allowed to specify what type of access can occur to the objects they own. DAC policies are based on the recognition of subject. In this model a user has full control over all the programs it owns and executes and also determines the permissions other users have those files and programs. Because DAC requires permissions to be assigned to those who need access, DAC is commonly called described as a "need-to-know"

access model and commercial operating systems typically enforce this model of access controls. The systems that use discretionary access control model grant or deny access based on the identity of the subject. This identity can be either a user identity or a group membership. Thus, for instance, a data owner can choose to allow Mr. X (who is a user identity) and the Accounting group (which is a membership identity) to access his files [16].

Mandatory access control

In mandatory access control (MAC), it is the system (and not the users) that specifies which subjects can access specific data objects. The MAC model is based on security labels. Subjects are given a security clearance (secret, top secret, confidential, etc.), and data objects are given a security classification (secret, top secret, confidential, etc.). The clearance and classification data are stored in the security labels, which are bound to the specific subjects and objects. When the system is making an access control decision, it tries to match the clearance of the subject with the classification of the object. For example, if a user has a security clearance of secret, and he requests a data object with a security classification of top secret, then the user will be denied access because his clearance is lower than the classification of the object. The MAC model is usually used in environments where confidentiality is of utmost importance, such as a military institution. Examples of the MAC-based commercial systems are SE Linux and Trusted Solaris.

Role Based Access Control

Role-based access control (RBAC) was introduced by Ferraiolo and Kuhn and became the predominant model for advanced access control. In RBAC model, the system administrators create roles according to the job function that are preformed in an organization, then grant proper permissions to those roles and finally assign users to the roles on the basis of their specific job responsibilities and qualifications [40]. Using a centrally administrated set of controls, RBAC model determines how subjects and objects interact, it also uses specific rules that indicate what can and cannot happen between a subject and an object [16].

The RBAC model is the best choice for the companies with high volume of turnover. For example if Mr. X who is working as a contractor in Alpha Inc., leaves the company, then Mrs. Y,

his replacement, can be easily mapped to this role. Using this model can save administration time for continually changing the ACLs on the individual objects. In this case the administration only need to create a role (e.g., contractor) and then assign permissions to this role and map the new user to the existing role. We can conclude that RBAC approach simplifies access control administration and can save lots of manpower time by allowing permission to be managed in terms of user job role [16].

2.5.2 Existing methods and models

In this section, we study existing security architectures and methods for controlling the access to data and information in multi-tenancy cloud environment.

Access control in cloud environments

In a cloud computing environment, it is utmost important to verify the identity and the access privileges of the service consumers before they are allowed to access the various resources or services hosted by the service providers. For preventing the unauthorized or illegal access of the cloud resources, it is extremely important to authenticate the requesting users and authorize their access privileges. So, in a cloud computing scenario, the access control of distributed resources is very vital in securing the cloud.

In this thesis, we studied different access control methods for cloud computing environments. For instance:

- Distributed access control (DAC) proposed by [41]
- Adaptive access algorithm proposed by [42]
- CloudPolice [43]
- Multi-tenancy based access control model (MTACM) [44]
- Role-based multi-tenancy access control (RB-MTAC) [45]

In the following sections, we will go through all the above-mentioned models and architectures to find out what are the existing access control models in cloud computing environments.

Thomas et al. [41] proposed an architecture for the distributed access control (DAC) in the cloud computing paradigm that three major components as bellow:

1. Cloud service provider (CSP)
2. Cloud service consumer (CSC)
3. Identity provider (IdP)

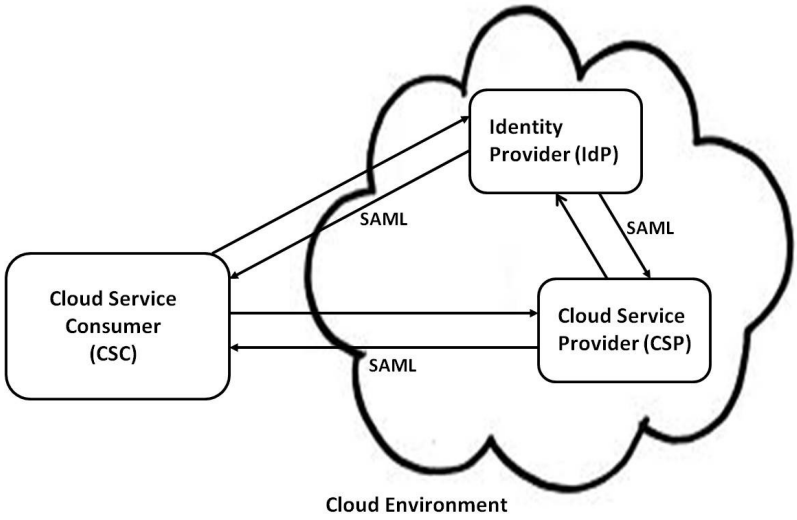

Figure 2-7 Architecture for the access control in cloud environments [41]

As illustrated in Figure 2-7, cloud service consumers (CSC) request the resources or services hosted by the cloud service providers (CSPs). In this stage, the CSC should be authenticated first to ensure that unauthorized users do not access the services from the CSP. The main responsibility of cloud service provider (CSP) is to host and to provide the various services or resources to the cloud service consumers. So, for avoiding illegal and unauthorized access by CSCs, proper authorization and authentication of CSCs are required. A user or an organization in a cloud environment may subscribe to services from multiple CSPs to meet the resource requirements; in this case a federated identity management approach (such as Single Sign-On authentication) is required. The CSCs can use the identity tokens generated by the Identity Providers (IdP) and these cloud users can exchange the security tokens generated by the Identity Providers with various Cloud Service Providers in the federation [41].

We can summarize the workflow of the access control architecture proposed by Thomas et al. [41] as following:

(i) CSC aims to access services hosted by the CSP,

(ii) CSP sends the authentication request to CSP,

(iii) CSC interacts with IdP to get the security tokens,

(iv) CSC interacts with CSC using the token issued by IdP,

(v) CSP authenticate the CSC by interacting the IdP,

(vi) The response to access request is communicated to the CSC.

Analysis and results of the above security architecture reveal that using a distributed access control (DAC) is very important in the domain of distributed application or services computing. At the same time this model has some limitations, since there is no single effective mechanism for DAC, which meets all the access control requirements of the Cloud computing domain, more researches must be carried out in that area.

Wang et al. [42], suggest an adaptive access algorithm by introducing the **trust** into cloud computing. Their aim is to solve more complex and difficult problems in the cloud computing environments by deciding the access control to the resources using an improved BRAC technique. The proposed security model determines dynamically security level and access control for the common resources. Thus, this architecture is supposed to provide appropriate security services according to the dynamic changes of the common resources. The proposed access control model is based on trust and determines whether the user has the right to get access to the resource by dynamically authorizing. This access control model claims that it can effectively control the user's malicious behavior.

In fact, M. Blaze proposed the concept of trust management in 1996. But Wang et al., they just took advantage of this concept and they added the concept of trust into role based access control (RBAC). This trust model is based on loyalty; it means that a user is restricted only when its behavior contains malicious behavior. In other words, a user can get access to resource or not dynamically based on a calculation. Wang calculates the user's trust by user's behavior when it successes or fails and base it to attribute user's right.

There is a difference between traditional access control mechanisms and trust-role-based dynamic access control mechanisms and it is the fact that the user's role can control policy by its respective trust level. Thus, trust is an important element when assigning roles to the users in a trust based access control model. This model is able to control the user's access to the resource orderly through establishing the dynamic mapping between roles and trust values.

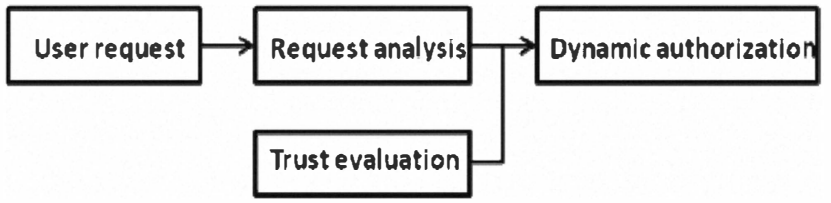

Figure 2-8 Dynamic authorization process based on trust in cloud computing

Figure 2-8 illustrates the process of dynamic authorization in a trust-role-based access control. In this process, the user's request should be analyzed first and the based on the trust evaluation, the user become authorized dynamically. The advantage of trust-role-based-access control approach are to extend the trusted computing technology into the cloud computing environment to achieve the trusted computing requirements for the cloud computing and then fulfill the trusted cloud computing. At the same time, this model needs to be developed more in order to become widely used.

Lucian Popa et al. [43], proposed CloudPolice, a system that implements a hypervisor-based access control mechanism. In 2010 they examined some types of access control (AC) policies (listed bellow) that they believe are important for multi-tenant clouds and from these examples they derived policy model and the used that policy model for design of their proposal. These access control policies are shortly explained bellow [43]:

- Tenant Isolation

  This access control policy is the most common and simplest prevention method for DoS attack. Tenant isolation simply prevents host from being compromised. This access control policy is traditionally implemented by VLAN (Virtual Local Area Network) but it is not a good solution for cloud environments.

- Inter-Tenant Communication

  Cloud computing offers a shared environment that will enable users to offer each other services more easily than with traditional business models.

- Fair-Sharing among Tenants

  Fair sharing is a concept that prevents tenants with more machines or higher available bandwidth from getting better service. Fair-sharing also prevent these tenants from impacting the services available to other tenants more than their fair-share.

- Rate Limiting Tenants

  Cloud computing provide us the “pay-as-you-go” pricing model. Some malicious agents may utilize this pricing model and they may financially damage their victims by increasing the bandwidth usage of each VM being attacked.

- Allowing Locally Initiated Connections

  Stateful firewalls can be implemented into traditional networks. Using that firewalls we can permit incoming traffics that are initiated from inside the network. Unfortunately this feature is not available in current cloud provider APIs.

Using the above examples, Lucian Popa abstracted a general AC policy model to be supported by cloud providers. They choose hypervisor-based approach for implementing CloudPolice policy model because hypervisors are trusted, network-independent, close to VM and fully software programmable.

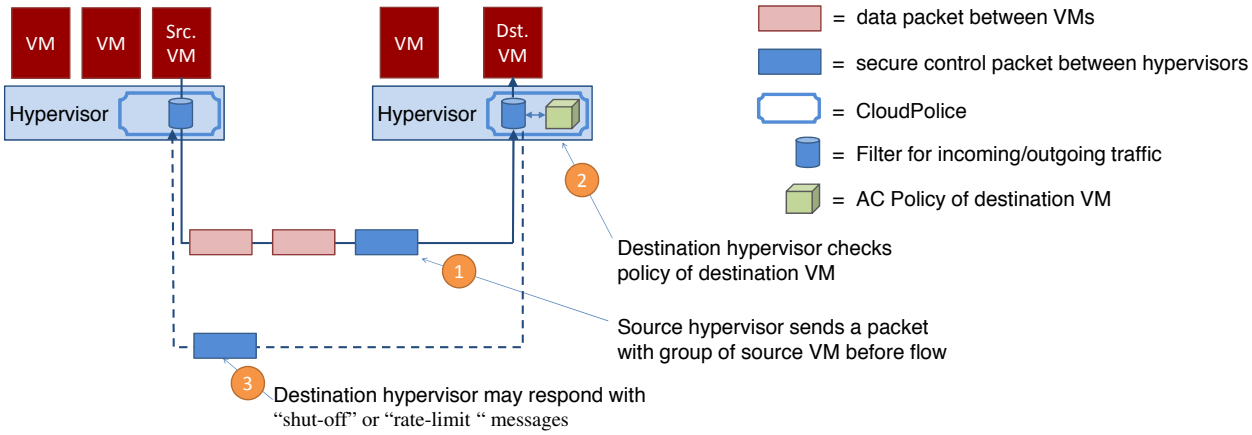

Figure 2-9 CloudPolice overview [43]

As demonstrated in Figure 2-9, when source VM initiate a new flow, the source hypervisor sends a control packet to the destination hypervisor. This control packet specifies the security group to which the source VM belongs (Step 1). As soon as the control packet reaches the source hypervisor, it will be checked by destination hypervisor to verify the policy for the group of the destination VM (Step 2). If the policy allows the traffic, then the state of the traffic will be created for this flow by the destination hypervisor. But if the traffic is not allowed or should be rated limited, the control packet will be sent back to source hypervisor to block or rate-limit the flow or the VM (Step 3).

As mentioned before, CloudPolice acts somehow as stateful firewalls and create state for each flow. Having this in mind, there are two major concerns for the feasibility of CloudPolice. First concern is the ability of hypervisor to act on per flow state because we explained before that CloudPolice is a hypervisor-based AC policy. So, the hypervisor should be ready to act on every single flow. The second concern is the ability of hypervisor to install new state with low enough latencies for new traffic flows. At this moment that we are conducting this research, only prototype of CloudPolice is implemented using the Open VSwitch. So we should make sure that hypervisor is able to create state for each new incoming flow very fast. In an experiment environment the number of VMs in hypervisor are limited but in a real cloud computing environment the number of VMs can be quiet a bit. Thus, the hypervisor should be able to create state for all new flows without latency (or at least with acceptable latency) and also act on the existing state that already exist on the buffer.

Xiao-Yong et al. [44], in 2010 designed Multi-tenancy based access control model (MTACM) which is a security architecture to embed the security duty separation principle in multi-tenancy cloud environments. MTACM is a two-granule level access control mechanism. One is tenant granule for CSP to compartmentalize different users and customers, the other one is application granule specified for customers and users to control the access to their own application. Limiting the management privilege of CSP and letting the customers to manage the security of their own business is the main idea of MTACM. Beside that, the aim of this security architecture is to help CSP to separate customers and make them secure their own applications and data [44]. Nginx, which is an open source proxy, was chosen for the implementation base of MTACM. Nginx enables developers to add extra security modules to provide value-added features.

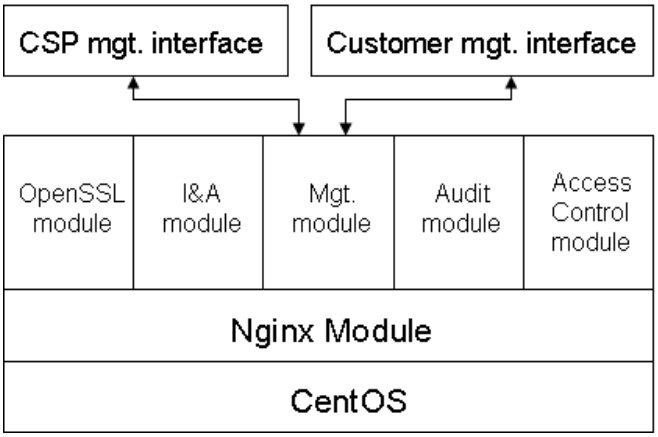

Figure 2-10 A prototype of MTACM [44]

As illustrated in Figure 2-10, there are five functional modules and two management interfaces. Functional modules are listed below [44]:

- OpenSSL modules – is an Nginx built-in module, which is activated to protect the confidentiality and integrity of communication during customers, access their cloud services in MTACM.
- Identification and Authentication modules –

- Audit modules – to record all the activities of security management and all accesses to cloud.
- Access control modules - to enforce DAC and MAC mechanisms.
- Management modules – to support the security duty separation mechanism between CSP and customer. This module has two separate interfaces:
    - CSP Management Interface (management control suites such as tenant adding, tenant removing and tenant management appointment are located in this interface)
    - Customer Management Interface (management controls such as user adding, user removing, object adding, object removing and ACL management are located in this interface)

We mentioned before that MTACM aims to separate security responsibilities between cloud service providers and customer. This security architecture, defines different management domains for CSP and customers. It means that customers should take the responsibility of managing and controlling their objects on the cloud.

Currently, most CSP perform customer compartmentalization and application access control by deploying firewalls [46, 47]. Figure 2-11 shows that different instances running on the same physical machine are isolated from each other via Amazon Web Service (AWS) firewall that resides within the hypervisor layer, between the physical network interface and the instance's virtual interface. But firewalls are not designed for this type of tasks and a firewall cannot handle this job because they do not use fixed IP address to access cloud resources.

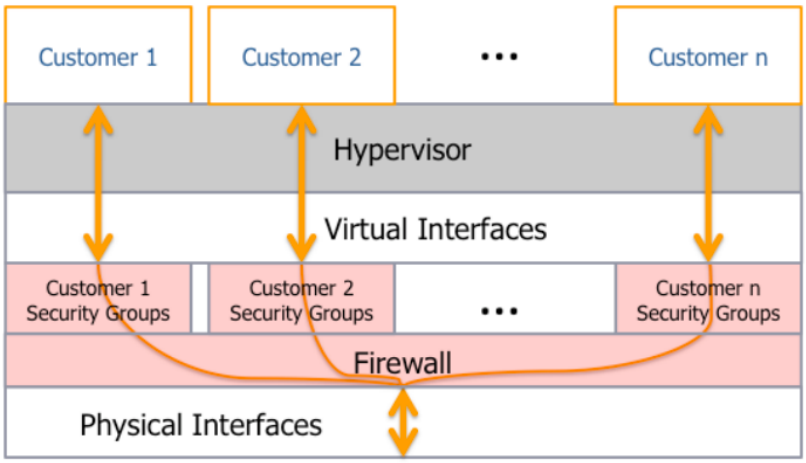

Figure 2-11 Instance isolation using firewall [47]

2.6 Problem and challenges

In this chapter, we performed a literature review in order to find out the works that has been done in the field of access control in cloud computing environments. Particularly, we discussed the existing methods of controlling access in context of multi-tenancy cloud computing.

We first considered distributed access control (DAC) architecture that was in the cloud computing paradigm [41]. This security architecture has three major components such as cloud service provider (CSP), cloud service consumer (CSC) and identity provider (IdP). In this architecture, the identity provider plays a great role because it should generate the identity token for users. By using this identity token, a user can request the access to the cloud. The main challenge with this architecture is that, there is no single effective mechanism for the DAC, which meets all the access control requirements of the cloud computing domain.

Furthermore, we've studied the adaptive access algorithm that is based on introducing the **trust** into cloud computing [42]. In this model the purpose of introducing the trust into cloud computing is to decide the access control to the resources. Wang et al. believe that using an improved RBAC technique they can solve more complex and difficult access control problem in the cloud computing environment. Trust base access control model uses trust as an important basis when assigning roles to the users. In other words, by establishing the dynamic mapping between roles and trust values, this model is able to control the user's access to the resources. This model has some advantages like extending the trusted computing technology into the cloud

computing. The main disadvantage of this model is that, it depends on the trust values and unfortunately the trust evaluation process is not developed yet and it needs to be improved more.

Then we studied multi-tenancy access control model (MTACM) [44] which is security architecture to embed the security duty separation principle in multi-tenancy cloud environments. The main idea of MTACM is based on limiting the management privilege of CSP and letting the customers to manage the security of their own business is the main idea of MTACM. In this security model, the duty separation mechanism between cloud service provider and cloud customer is handled by management module. In fact, the main drawback of MTACM is this management module. Because this management module is not user friendly for customers and the cloud customer has to take care of the security of their data.

For complimentary, we reviewed role-based multi-tenancy access control scheme for cloud services [45]. The RB-MTAC applies identity management to determine the user's identity and applicable roles. In fact, this RB-MTAC combines tow important concepts in access control under the MTA environment, these two concepts are the identity management and also role-based access control. Yang et al, they believe that this combination makes it easier to manage privileges that protect the security of application systems and data privacy. Providing a set of privileges and also identity management scheme for corporations in cloud computing environment is the main contribution of this security model. This scheme can be used to easily change employee privileges in the when personnel leave an organization or when we want to grant more access to them without the need to modify all employee privileges on e by one. But the main disadvantage of RB-MTAC is that it is not independent and for implementing this security model in a cloud computing system, a directory service is needed.

Finally, we reviewed CloudPolice [43], an interesting access control method for multi-tenancy cloud computing environments. This security architecture attracted our attention because is seems to be effective and can prevent denial of service (DoS) attack, which is one the major issues in cloud environments. This security architecture can prevent malicious agents from sending unauthorized traffic to their targets. But a careful study about CloudPolice reveals that it imposes some overheads into the system. In this model, the destination hypervisor receives all the traffic and decide to pass or drop the traffic based on the security attributes of the target virtual machines. In this model, every single flow should travel from source hypervisor to the destination

hypervisor and because of this fact; still a malicious agent can exhaust the bandwidth of the link between source hypervisor and the destination hypervisor.

Through this literature review, we studied possible algorithms and systems of controlling access in multi-tenancy cloud environments. This state of art enables us to build on this knowledge in order to build a solution of our own in the course of the forthcoming chapters.

CHAPTER 3 PROPOSED ARCHITECTURE

This chapter presents the architecture to improve the access control of the multi-tenancy cloud environments. Firstly, we will explore a scenario for our project, which has two main phases. Later on we will define the requirements of the security architecture that we want to design. Finally, we will explain the design and basis of the proposal; this part includes assumptions, principles, elements and topology of the architecture.

3.1 Scenario

In this section we will define a scenario and by doing that we will be able to tackle particular security issues. Evidently, this use case is in the context of cloud computing and especially in the context of multi-tenancy environments. We will analyze this scenario in order to define the most relevant answer to a particular problem that is unauthorized access of the VMs for sending unauthorized traffic to other VMs. As we mentioned before, this security architecture aims to secure the access control. So, use case is the key to the definition of our requirements.

This scenario is the simplest one possible and is illustrated in Figure 3-1. In this example, we have a public cloud that is connected to the Internet using router and three physical servers that are connected to a layer-2 switch. The function of the router in this scenario is to route the internal traffic of the cloud to the internet and vice versa. Apparently, the layer-3 switch in this scenario serves as a controller, enabling networked devices to talk to each other efficiently. In this scenario, there are 3 physical servers as well as 10 virtual machines. These virtual machines belong to 4 tenants. Multi-tenancy topology of this cloud is as bellow:

- Physical Server 1: Tenant 1 (VM1, VM2) and Tenant 2 (VM3)
- Physical Server 2: Tenant 1 (VM4, VM5) and Tenant 3 (VM6, VM7)
- Physical Server 3: Tenant 4 (VM8) and Tenant 3 (VM9, VM10)

For the purpose of load balancing, this cloud provider has decided to use the above topology. Load balancing is dividing the amount of work load that a physical server has to handle between two or more physical servers so that more work load gets done in the same amount of time and, in general, all users get served faster. That is why some VMs that belong to tenant-1 are located on the physical server 1 and the rest of them reside on the another physical server.

According the above explanation about the load balancing in virtual environments, VMs that belong to tenant 1 are located on the physical server 1 and physical server 2. Also for the same reason the virtual machines of tenant number 3 is distributed into physical server number 2 and physical server number 3. But the tenant number 4 has only one virtual machine that is located on the physical server number 3.

It is necessary to mention that in this scenario, the process of the controlling the access is done in the hypervisors. Apparently, the ACL (access control list) database resides on the hypervisors and once traffic is entering the hypervisor the relevant ACL for the traffic should be exists there.

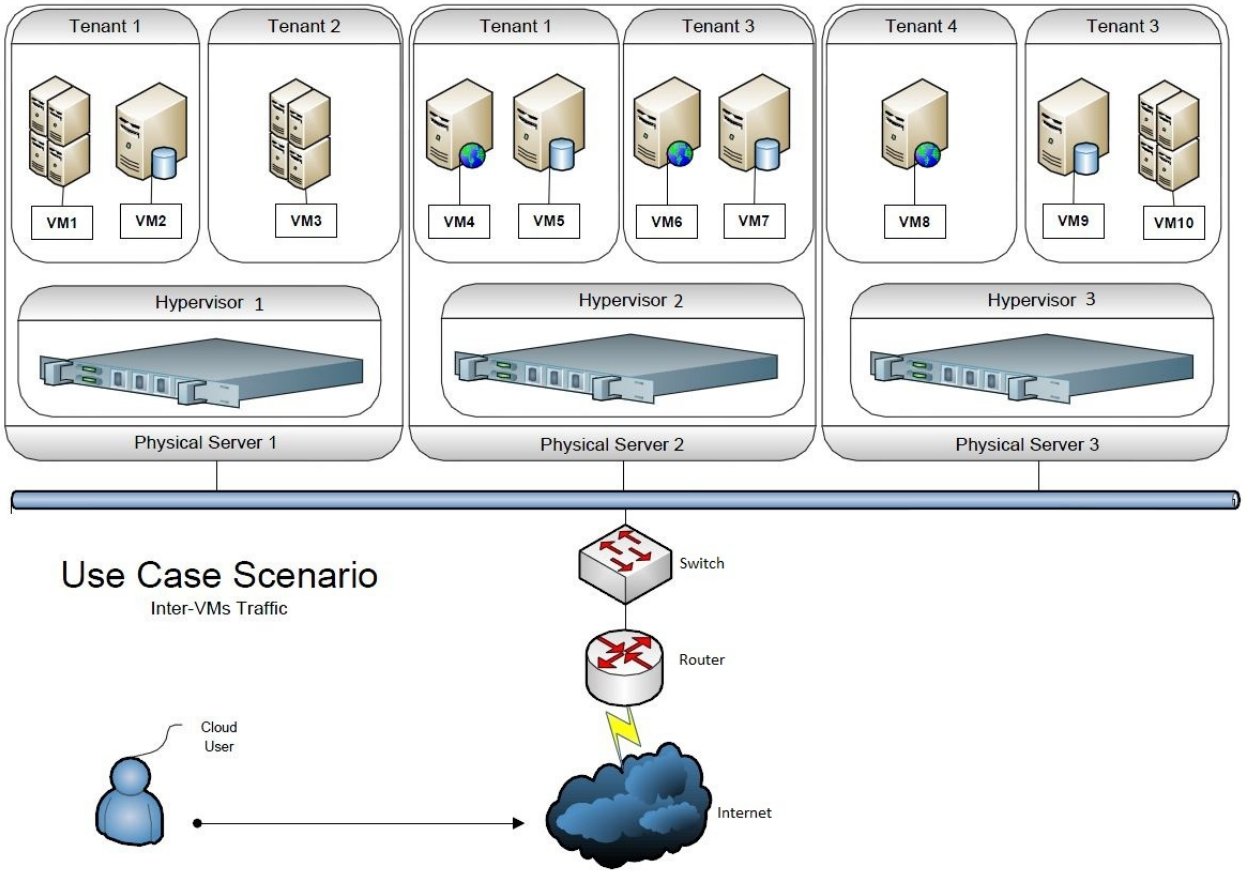

Figure 3-1 Use case scenario for multi-tenancy cloud access control

This scenario has two phases; the first phase consists of generating the control packets. This control packet contains information about the traffic that a source VM wants to send to a

destination VM. In the second phase, the destination hypervisor investigates the information and decides about the destiny of the packet. In the following section, we go through each phase in detail.

Phase One:

In the first part of the scenario, the VM Source sends a flow of traffic to the hypervisor source as it is illustrated in the stage 1 of the Figure 3-2. Obviously, the VM source and the hypervisor source are co-located on the same physical server. Then the hypervisor source generates a packet that is called *control packet[43]*. This control packet contains some information about the source address, destination address and port number. As illustrated in stage 2 of the Figure 3-2, when the control packet is generated based on the information in the data packet, the source hypervisor sends the control packet to the hypervisor destination in order to check the access control policy of the VM destination.

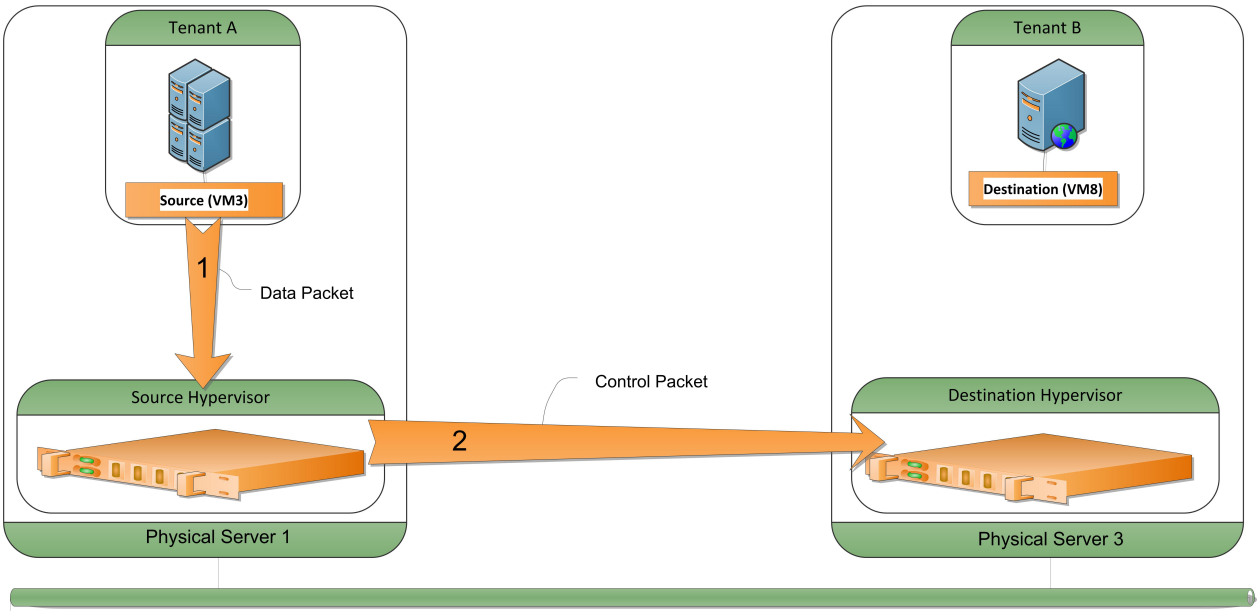

Figure 3-2 Phase one, generating control packet by source hypervisor

So, in the first phase of the scenario, the hypervisor source generates the control packet. The content of this control packet is based on the traffic that is to be sent from source VM3 to destination VM8. As soon as the source hypervisor generates the control packet, it sends this control packet to the hypervisor destination.

Phase Two:

In this phase, the control packet arrives to the destination hypervisor. Then the destination hypervisor checks the access control lists (ACLs) to verify if the VM source is authorized to send that traffic to the destination VM8. If the ACLs related to the destination VM8 match, then the destination hypervisor sends back a pass value within the control packet called response control packet to the source hypervisor as it is illustrated in the stage 3 of the Figure 3-3. The purpose of this response control packet is that, the hypervisor source can decide what to do with the traffic – that is waiting in the source hypervisor - according to this response control packet.

Hence, if the security attributes of the destination VM8 do not match the data packet, then the destination hypervisor sends a drop signal to source hypervisor.

In this phase, there can be different possibilities as bellow and our proposed architecture will be able to cover and work in any possible situation.

- Source VM and destination VM are located on the same physical server but belong to deferent tenant
- Source VM and destination VM are located on the same physical server and belong to the same tenant
- Source VM and destination VM are located on the different physical server but belong to the same tenant
- Source VM and destination VM are located on the different physical Server and belong to different tenant

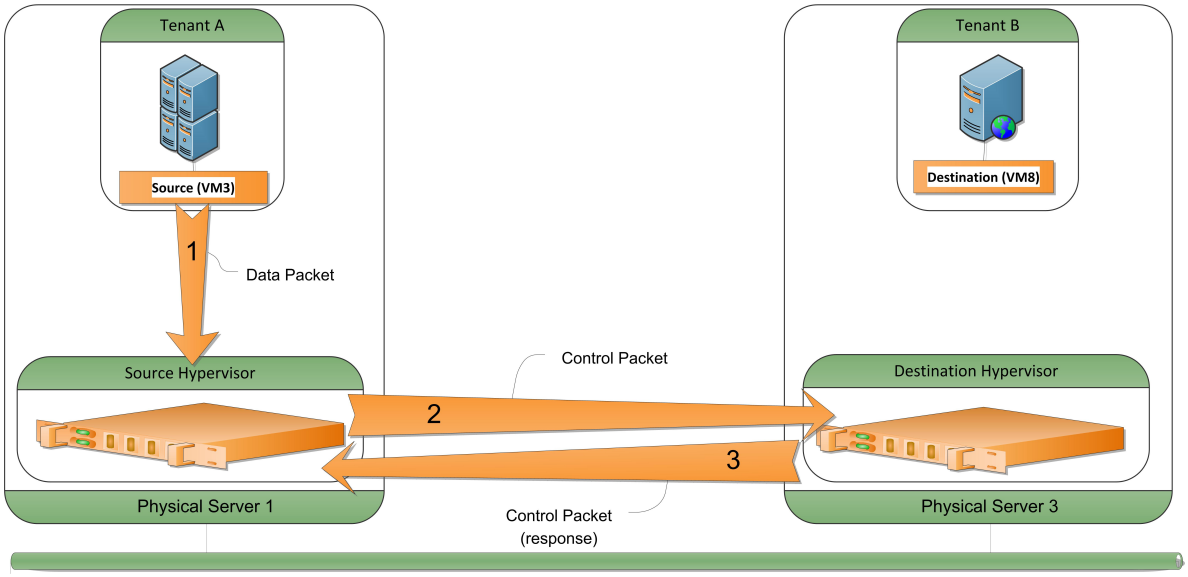

Figure 3-3 Phase two, generating control packet response by destination hypervisor

3.2 Requirements

In order to build and design architecture for securing access control for multi-tenancy cloud environments, we need some requirements and criterions. In the following section we describe the requirements of access control security architecture.

3.2.1 Scalability

Scalability is one of the characteristics of the cloud computing environments. So, the security architecture should be scalable enough to cover this growth rate. As reported by Amazon Web Service, Inc. (AWS), in 2013 ,each day near 100K new VMs started [43]. This report shows that the cloud computing infrastructure is huge; therefore, our security architecture should easily leverage this incredible infrastructure scale. In this scale, one VM may communicate with many other VMs in the cloud, so controlling the access of network entities to the network resources is a vital part of securing cloud infrastructure.

3.2.2 Security

The access control security architecture that is designed for multi-tenancy cloud environments should be able to maintain the security of information of all the tenants and users. In a multi-tenancy context, access of VMs should be controlled. So, a VM should only have access to other VMs if it is authorized. In our security architecture design, only authorized VMs

are able to send traffic to other VMs. If a VM sends unauthorized traffic to another VM, the source hypervisor drops the traffic packet.

3.2.3 Control

Controls are countermeasures to minimize the security risks and also to avoid them. Access control is an effective tool to protect the confidentiality, integrity and availability of information in a multi-tenancy cloud environment. This security architecture is designed to enforce the access control lists. In the traditional network environments, the traffic flow pass through the security devices such as firewalls or layer-3 routers. But in virtualized environments, the traffic passes through the vSwitch of hypervisor rather than security devices. So, the security architecture of access control should be able to enforce the ACLs in a way that, it prevent unauthorized access to the cloud.

3.2.4 Performance

By dropping the unauthorized traffics in the source hypervisor, we can avoid unnecessary traffic in the cloud environment. A malicious tenant in a cloud system can target another tenant and generates tones of unnecessary packets to consume the cloud system resources and make the cloud resources unavailable to for other tenants. So, if the security architecture prevents the unauthorized traffic, the bandwidth of the network can be dedicated only to authorized traffic and this can improve the performance of the network.

3.3 Architecture overview

The idea behind this architecture is that a virtual machine in a multi-tenancy cloud system should not be able to send traffic to another VM unless that virtual machine belongs to the same tenant and/or this VM is authorized to send the traffic to the other VMs. We propose a security architecture that prevents tenants of cloud computing environment to send unauthorized traffic to other tenants.

This security design deals with the concept of Inter-VM traffic, which is transmission of any data packet to and from one virtual machine. In the other words, when the hypervisor encounter Inter-VM traffic, the traffic does not pass through physical switch or router in a cloud system and the virtual switch that is located in the hypervisor forward the packet to the

destination VM. Once the architecture is fully presented in this chapter, then, a detail analysis on how the requirements are met by the proposal will be done in the next chapter.

3.3.1 Assumptions

The idea behind this architecture is securing access control in multi-tenancy cloud environments; however there are some assumptions that need to be done at this point as a way to present achievable goals at this stage in the design process.

- Concerning the entire system, we assume that entire virtual machines and physical servers are co-located and on the same cloud provider. If the entire system is not part of a cloud, then for sending traffic to another cloud, the traffic should pass through a real router or firewall. In this case we should enforce the policies that are in the firewall.
- Furthermore, we assume that each physical server on the cloud has only one hypervisor. In this case, security attributes and access control lists of all virtual machines that belong to a physical server are located in one hypervisor. If we have multiple hypervisors on a physical server, then we should apply an extra process for realizing which hypervisor contains access control lists of certain virtual machine.
- Additionally, we assume that each physical server is hosting at least one tenant and each tenant has at least one virtual machine. Because each virtual machine should be registered as a Tenant in the system. If a tenant is registered in the cloud, consequently a virtual machine should be assigned to that tenant, if not, that empty tenant can cause point of weakness in the cloud.
- It is very important to mention that the goal of this project is not to design or define the security access control policies, so we assume that all access control lists are defined and stored in the hypervisor. Therefore, it is very important to store the access lists in the hypervisor, not in the middle boxes.
- In order for our security design to work, each hypervisor should receive periodic updates about IP address of virtual machines that are located on entire cloud. So, we assume that in the start up process of a hypervisor, it sends an update message to other hypervisors that are located on the same cloud. This update message contains the IP address and the ID of virtual machines that are located on that hypervisor.

Having the above-mentioned assumptions in mind, we will explain the different aspects of this security architecture down the road to the rest of this chapter.

3.3.2 Architecture principles

The principles of the proposed architecture are based on control packet, which is the core element of verifying security permissions of virtual machines in a multi-tenancy cloud environment. Figure 3-4 depicts the general architecture of our security design and in this section we explain it in detail.

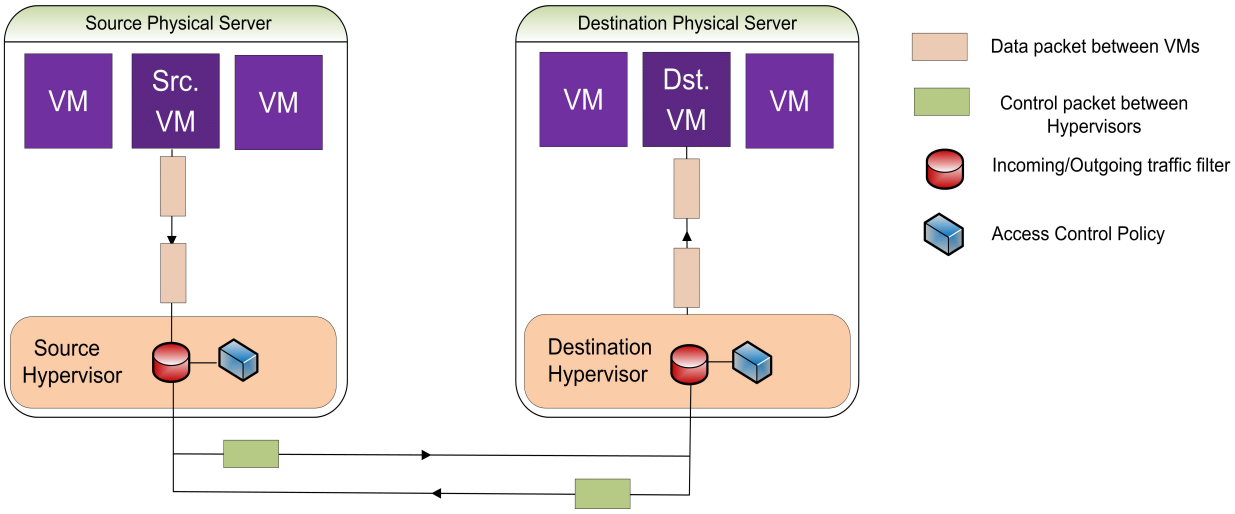

Figure 3-4 Principles of security architecture

- Source (Src.) VM is a virtual machine that is located on the source hypervisor. This hypervisor is located on the source physical server. Source VM is sending some traffic packets to another virtual machine in the same cloud that is called destination
- Destination (Des.) VM is located on the destination hypervisor and obviously this hypervisor is located on the destination physical server.
- Data Packet is the original packet that source VM wants to send to destination VM.
- Control Packet is a special packet that is generated by source hypervisor. The content of this packet represents the specifications of Source VM and the destination VM.

- Incoming/Outgoing traffic filter is a lightweight IDS that is integrated in the hypervisor. It simply compares the control packet with the access control lists of destination VM.
- Access control list is a set of security permission that defines the level of security of each virtual machine.

3.3.3 Control packet

Control packet is a set of bytes called frame tag that we add to the IP packet[43]. These bytes are inserted after IP header and at the beginning of the payload as it is shown below. The frame tag in our architecture has two main parts: Header of the first data packet of the flow as well as the action command value.

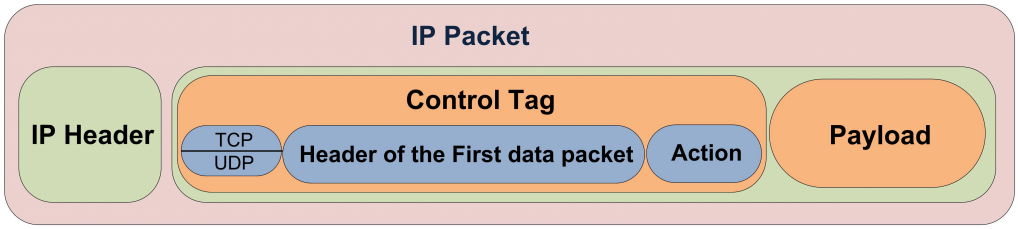

Figure 3-5 Control packet

TCP or UDP:

The first frame tag that is inserted into the control packet shows the type of Internet Protocol (IP) traffic. Due to the fact that there are two types of IP traffics (Transmission Control Protocol and User Datagram Protocol), this tag determines the type of IP traffic. If the traffic is TCP traffic, then the TCP tag will be inserted to this place. But if the traffic is UDP, consequently the UDP tag will be inserted to this place.

Header of the First Data Packet:

The second frame tag that is inserted into the control packet is Header of the First Data Packet. When a traffic flow arrives the source hypervisor, only the first data packet of that traffic flow is chosen and the header of this data packet will be inserted to the control packet. Although each IP protocol has a specific IP header fields, there are some fields that are the same between both UDP and TCP protocols[48]. Therefore, for this part of the frame tag, we use the value fields that are the same such as:

Source Address could be any class (A, B, C or D) of IP address that is assigned to the source VM by dynamic host control protocol (DHCP) service or manually.

Destination Address is the again an IP address that is assigned to destination VM.

Port Number is the Internet socket port number that is used by protocols of the transport layer of the internet protocol suite for the establishment of host-to-host communication [48].

Protocol here is whether transmission control protocol (TCP) or user datagram protocol (UDP) that is a transportation protocol and one of the core protocols of the internet protocol suite [48].

Action Tag

The action tag is a value that is inserted to the control packet by the destination hypervisor or source hypervisor and specifies what action should be applied to the traffic flow. Generally there can be three different values for the action tag.

- **NULL Value**: Initially, this action tag value is inserted to the control packet by source hypervisor when it generates the control packet. In fact, the action tag field is reserved for destination hypervisor. When the source hypervisor generates the control packet, the action tag should be left empty so that the destination hypervisor insert the proper value to this empty field. Consequently, when the destination hypervisor wants to send back the response control packet, it uses this field to insert the proper value. But, in some cases, if the destination VM is not found in the destination hypervisor, this value remains empty (the same NULL value). In this case, this empty value resembles no decision and the source hypervisor should take another extra actions for sending the control packet to another hypervisor in the cloud.

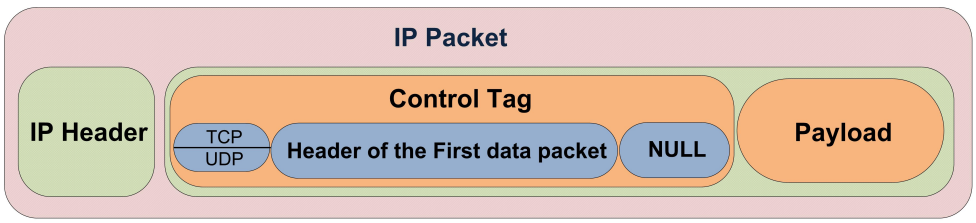

Figure 3-6 Empty or null action value

- **PASS Value**: This action tag value is inserted to the response control packet by the destination hypervisor. The PASS value means that the traffic is an authorized traffic. When the destination hypervisor receives the control packet, it checks the security attributes of the destination VM with the Header of the first data packet, if access control list allows the traffic, the destination hypervisor insert PASS value to the response control packet. There fore, this authorized traffic can be passed to destination hypervisor. As soon as the source hypervisor receives a response control packet with PASS value, it passes the traffic flow to the destination hypervisor.

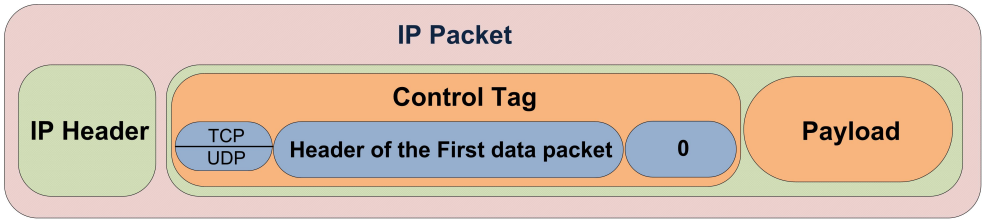

Figure 3-7 Pass action value

- **DROP Value**: Like the previous one, this action tag value is inserted to the response control packet by the destination hypervisor too. DROP value means that the traffic should be dropped right on the source hypervisor. The destination hypervisor insert this value to the response control packet when the security attributes of the destination VM do not match with the traffic flow.

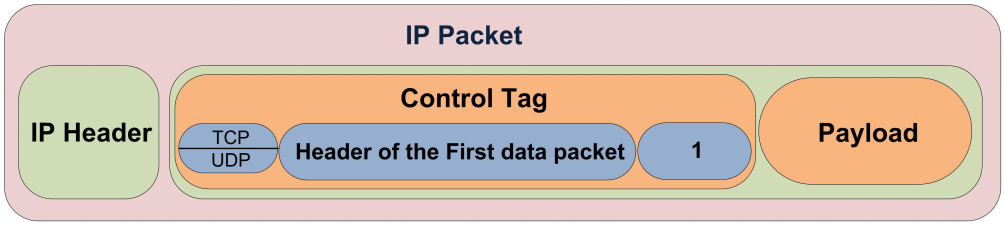

Figure 3-8 Drop action value

3.3.4 Traffic filter

Filtering the traffic is a major part of this security design. After generating the control packet by the source hypervisor, it should be sent to the destination hypervisor. In this part, the destination hypervisor should filter the traffic and send the response control packet back to the source hypervisor.

Filtering the traffic is done by the use of ACLs. In fact, in this security architecture, access control lists of each VM are located in the hypervisor where the VM is located. Thus, access control list is an essential part of multi-tenancy cloud environments in order to prevent tenants compromising virtual machines of other tenants. There are many different ways that a malicious VM can compromise another VM. A cloud user can perform different types of attacks –such as denial of service- against another user. It is good to mention that according to [49] the existing mechanism for securing the multi-tenancy cloud environments are inherited from enterprises. Two of those mechanisms are:

- VLANs
- Firewalls

The above mentioned traffic filtering mechanisms are initially designed for enterprise networks that have fewer requirements for flexibility and scalability and also computing resources in enterprise networks are shared between fewer distinct organizations. In other words, in an enterprise network, rarely one user tries to perform attack (for example DoS attack) to another user.

Unlike enterprise networks, cloud networks have a network design that is different in the sense of scale. It means that the cloud networks are usually larger than enterprise networks. Because of this fact, for designing access control security architecture for multi-tenancy cloud networks we should always take the following characteristics into consideration:

- Scalability of design
- Flexibility of policies
- Multi-tenancy nature of cloud

Although the AC policies are defined, set and stored by the destination, it's better for the policies to be enforced close to the source rather than at the destination. This fact can vastly prevent unauthorized traffic from abusing the network. Unauthorized traffic can cause congestion and also DoS attacks and etc. [43]

The prototype of proposed architecture is based on two clear and distinct access control policies:

- **Pass** the traffic
- **Drop** the traffic

Basically, these two actions are done on the source hypervisor. This decision is made based on the control packet response that source hypervisor receives from destination hypervisor. Beside Passing and Dropping the traffic, there are more possible actions that could be taken like limiting the traffic and so on, but we leave those for future work.

3.3.5 Architecture design

As illustrated in Figure 3-4, when the source VM sends the traffic to destination VM, the traffic has to pass through the hypervisor that it is located on it (we call it source hypervisor). As soon as the data packets reach the hypervisor, it generates a control packet. This control packet consists of necessary information for access control checking as well as source IP address of the packets, destination IP address of the packets, port number and protocol type (TCP or UDP). So the control packet has to be sent to destination hypervisor. Destination hypervisor checks the content of the control packet and decide whether the traffic can be delivered to destination hypervisor or not.

If the source VM is permitted to send the so-called traffic to the destination VM, so the destination hypervisor adds the pass or drop value to the control packet payload and sends it back to the source hypervisor. According to this pass or drop values, the source hypervisor threat the awaiting traffic.

Detailed design

Our proposed architecture is all about providing *network access control* between the virtual machines in the context of multi-tenancy cloud computing. The ultimate goal of this architecture is to block and drop undesired packets as close as possible of source VM that in our case is source hypervisor.

As you can see in Figure 3-9, the process starts when a VM initiate a traffic to send to another VM, as soon as traffic received by the source hypervisor, it checks the packet and looks

for the destination address that is located in the inserted in the IP packet header. If the destination address belongs to a virtual machine in the same cloud, we will have two possibilities:

- First Case:

  Destination address is located on the same physical server (hypervisor). In this case, our security architecture has a simpler task to do. It has to only check the access control policy of the destination VM –that is located on the same hypervisor- and it can decide right away whether to Pass or Drop the traffic.

- Second Case:

  Destination address is located on different physical server (hypervisor).

  In this case, the source hypervisor generates and sends the control packet to the destination hypervisor where the destination VM is located. Then it should wait for the response control packet from the destination hypervisor.

Beside the so-called possibilities that we mentioned before, there may be an exception. This exception occurs when the destination address does not belong to any VM in this cloud. It means that source address and destination address belong to two devices that are not co-located in the same cloud. In this case, this architecture does get involve into doing any particular action. It only has to pass the traffic to the middle box (router, switch, firewall or…) that is default gateway of the source hypervisor.

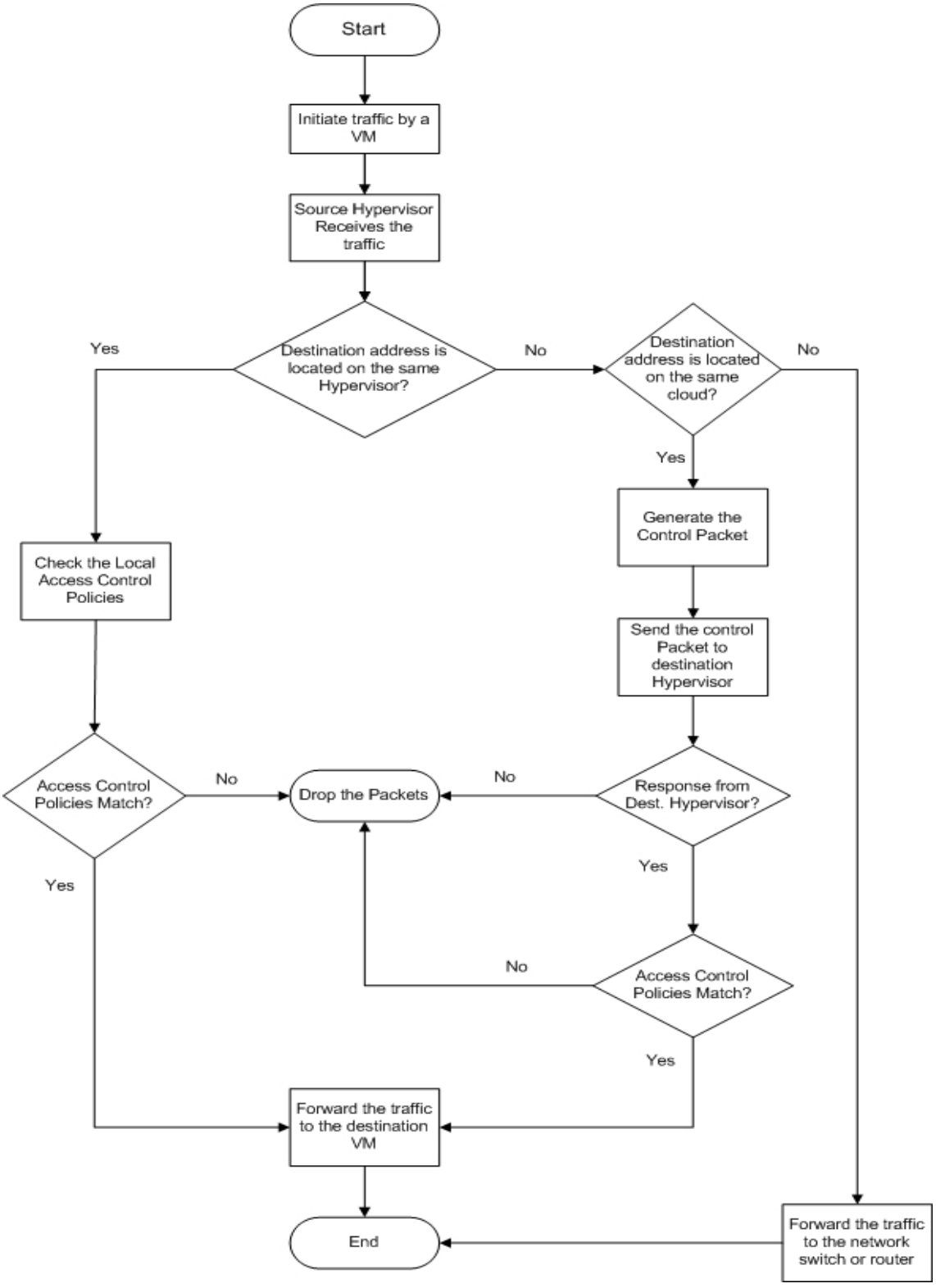

Figure 3-9 General mechanism flowchart

The main part of mechanism starts if the destination address belongs to another VM that is located on another hypervisor (destination hypervisor). In this case, the whole traffic should wait until the source hypervisor generate and send a control packet to destination hypervisor. Hence, the decision will be made based on the response control packet.

Destination hypervisor

Figure 3-10 shows the main task of destination hypervisor when it receives the control packet from source hypervisor.

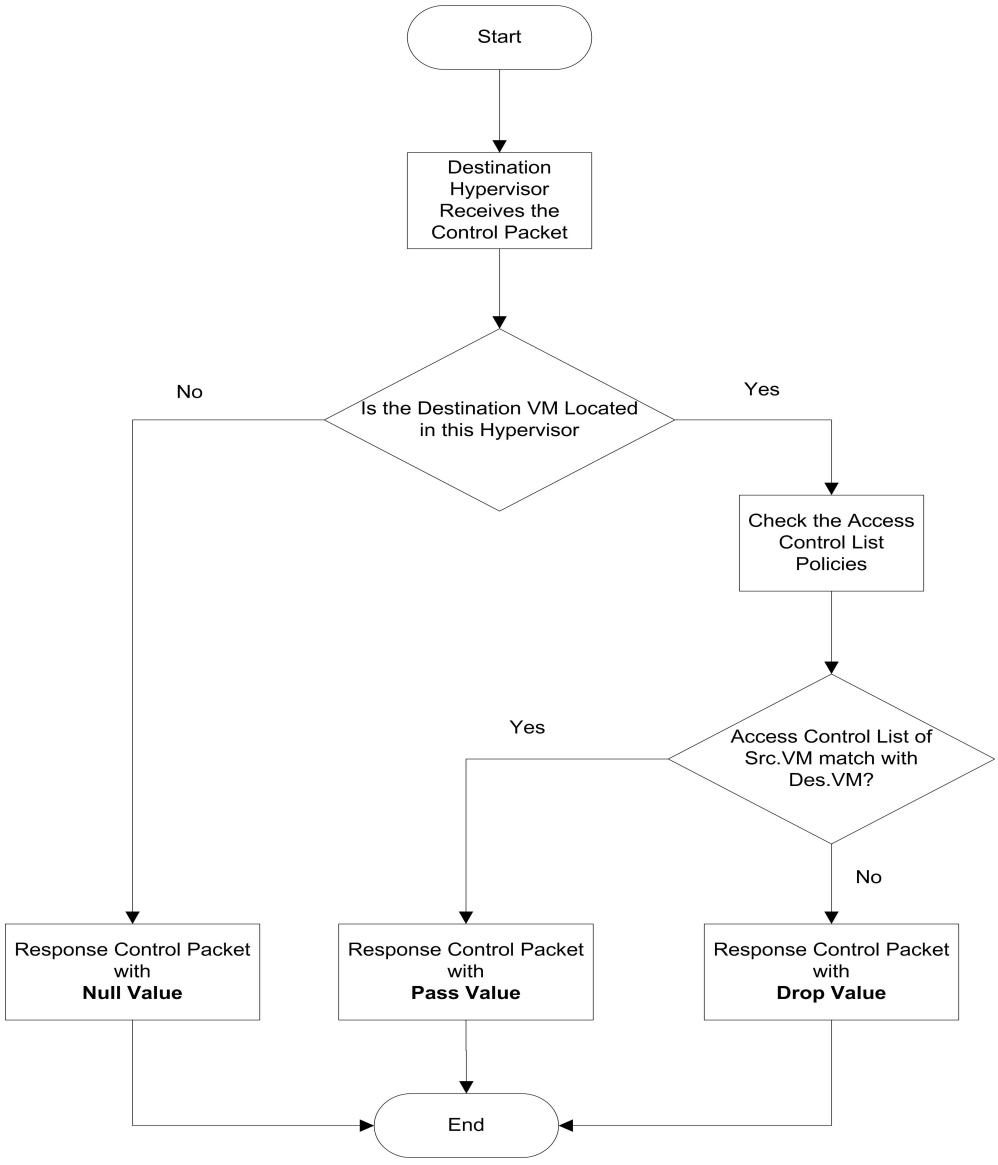

Figure 3-10 Destination hypervisor's task when it receives a control packet

In fact, destination hypervisor has to select one of the following options when it receives the control packet form the source hypervisor:

- **Insert pass value** to the control packet if the access control policy of the destination Virtual Machine match, and accept the traffic from source VM.
- **Insert drop value** to the control packet if the access control policy of the destination VM does not match and source VM is not authorize to send the traffic to the destination VM.
- **Insert null value** to the control packet of the destination address is not found in the destination hypervisor. This may happen if control packet is sent to this hypervisor by mistake or is destination VM is migrated to another hypervisor and source hypervisor is not updated about the VMs that are located in the destination hypervisor.

After performing one of the above actions and inserting proper value to the control packet, the destination hypervisor returns the edited control packet to the hypervisor that generated the control packet, which in our case is source hypervisor.

Source Hypervisor

The response control packet, most likely contains the decision and the action that should be taken for the traffic. Why most likely? Because if the source hypervisor sees the Null value in the control packet, it knows that destination VM is not found yet and it should regenerate the control packet and send it to another hypervisor. This step should be done after asking all other hypervisors for the update message. If the update massages are all received and the destination VM is not found, so the traffic should be dropped consequently (Figure 3-11).

In the case of drop value, the source hypervisor drop the traffic right away and the traffic will not even exit the hypervisor. It is great, isn't it? It means that no wasted and unnecessary traffic in the network, and consequently the network bandwidth do not suffer from extra and unwanted traffic. In the next chapter, we will try to implement the architecture and see the result.

Finally the pass value indicates that the access control policy match between the source and destination and source VM and the traffic will pass to the destination hypervisor.

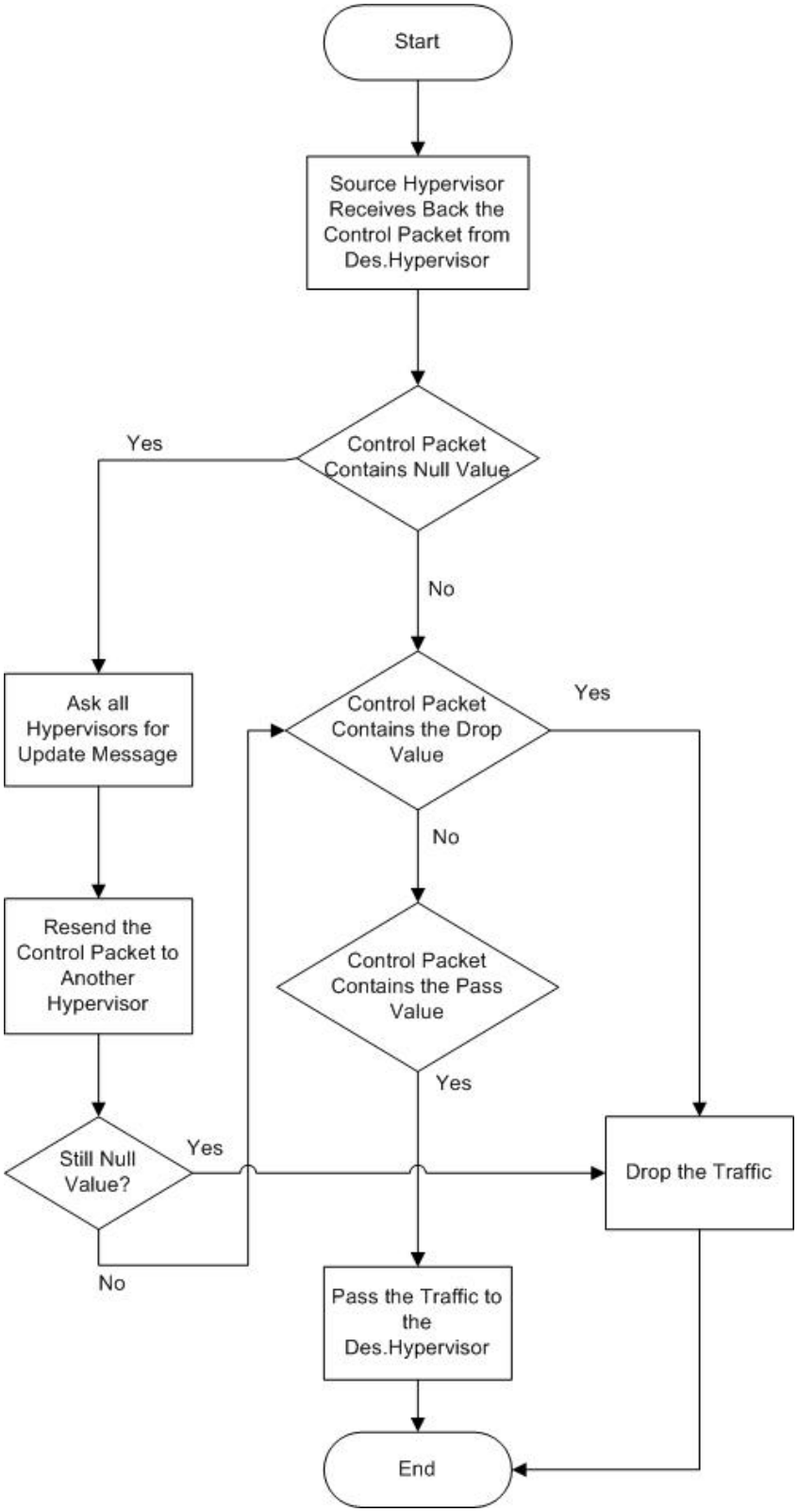

Figure 3-11 Source hypervisor's task when it receives back a control packet response

Work Space

In order to show the results of this project we will implement the prototype of this architecture using an open source virtualization system. Among the existing virtualization tools we can name Xenserver and KVM (Kernel-base Virtual Machine). But in this project we use Xenserver because it supports full virtualization and almost all existing operating systems.

Beside that we use OpenStack, an open source cloud computing project to provide an infrastructure as a service (IaaS). OpenStack was prompted in 2012 and the technology behind it consists of a series of interrelated projects that control pools of processing, storage, and networking resources throughout a datacenter, all managed through a dashboard that gives administrators control while empowering its users to provision resources through a web interface.

Sequence Diagram

Sequence diagram (Figure 3-12) presents a dynamic view of the proposed architecture. As mentioned in the use case description, this system has two main phases, the first phase is the process of generating the control packet which is nothing but inserting the control tags into the IP packet and the second is the process of sending this control packet to the destination hypervisor and based on the response control packet, the source hypervisor decides about destiny of the traffic. This sequence diagram illustrates both phases along with the interaction of the four main elements of a cloud network, which are source VM, source Hypervisor, destination VM and finally destination hypervisor. In this illustration we marked the Pass Value of control packet response as green, which means that the traffic is permitted to forward to destination address. Obviously, when the source hypervisor receives the Drop Value from destination hypervisor it indicates unauthorized traffic and consequently the traffic has to be dropped right away.

This process becomes somehow more complicated if the hypervisor receives the NULL Value from destination hypervisor. In this case the source hypervisor should regenerate the control packet and send it to another hypervisor till the destination VM is find or make sure that the destination address is either wrong or not found in the network. In all cases of NULL Value, the traffic should be delivered to destination VM or be dropped.

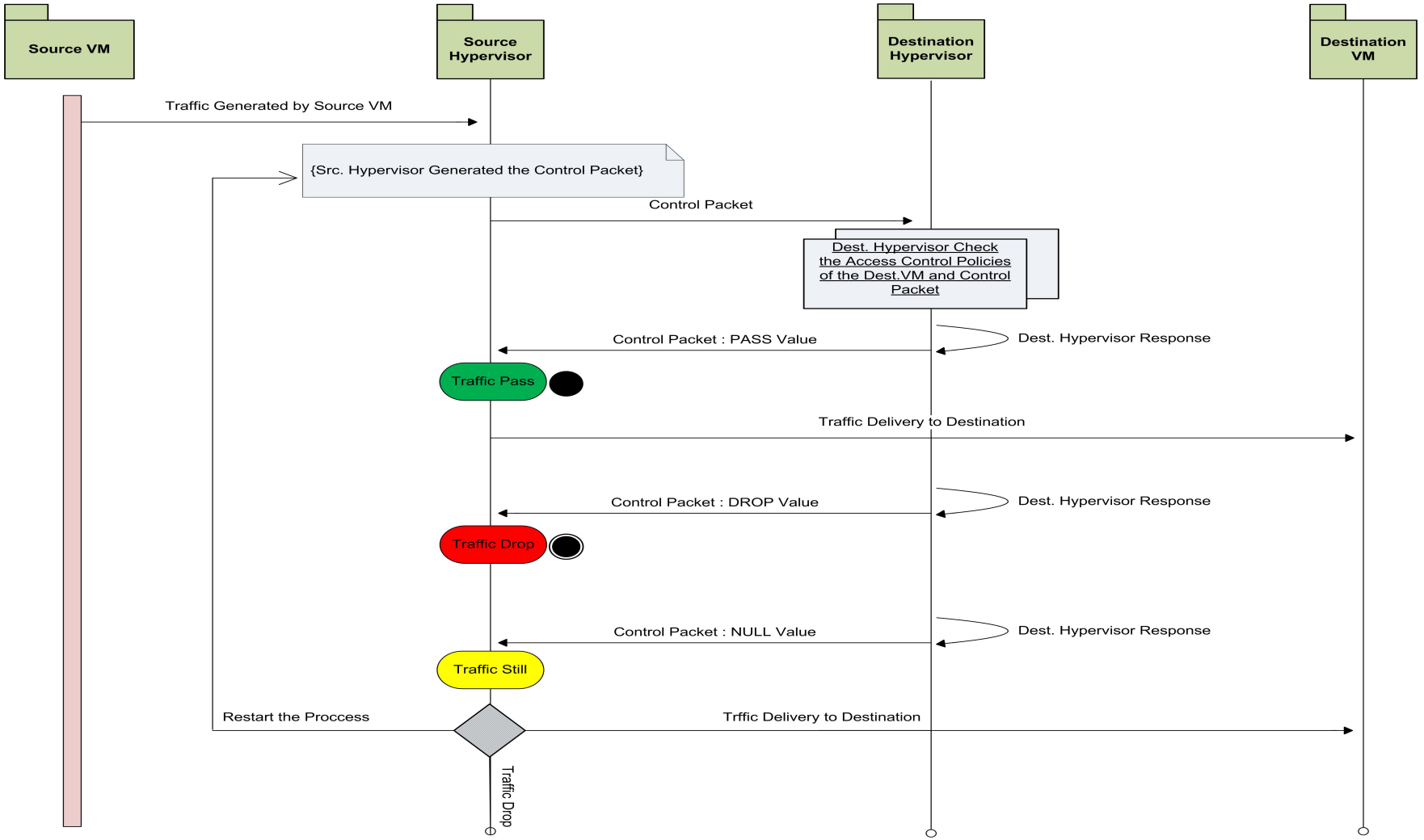

Figure 3-12 Process sequence diagram

In some cases (mentioned in phase two of use case scenario), the destination address belongs to a VM, which is located on the same hypervisor. In this situation we can have two different possibilities:

A: Source and destination VM belongs to the same tenant

B: Source and destination VM belong to different tenants

In any of the above cases, the source and destination hypervisor are the same. So, because the hypervisor can directly check the access control list of the destination VM, therefore, the hypervisor does not generate any control packet (Figure 3-13). Obviously it can decide about the destiny of the traffic by itself. In this case we don't have the possibility of the NULL Value too.

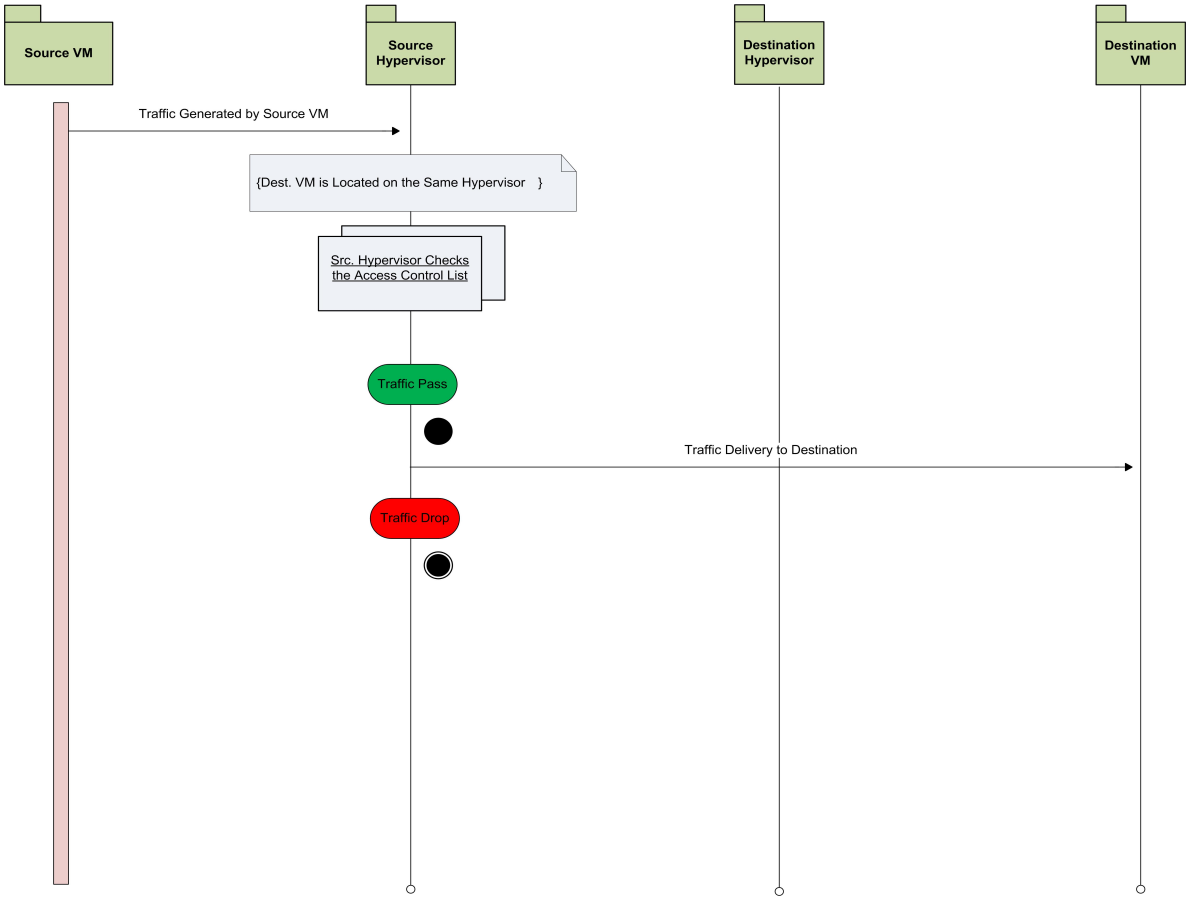

Figure 3-13 Source VM and destination VM on the same hypervisor

3.4 Conclusion

In this chapter, we explained our proposed access control architecture for multi-tenancy cloud environments in detailed. Also we defined a scenario and based on that scenario we investigated the problems and requirements. In order to solve the problems and satisfy the requirements, we designed our security architecture, and using a set of illustrations and flowcharts, we explained the architecture in depth. As we mentioned earlier in this chapter, there are four main requirements for this security architecture such as scalability, security, control and performance. This security architecture is fully scalable. If the number of VMs grows, we only need to implement this architecture in the hypervisor of each physical server without any extra changes in the system. Besides that, this security architecture can maintain the security of information in the cloud system. This can be achieved by our security design when it controls the traffic that is sent from one hypervisor to another hypervisor and by enforcing the security policies in the hypervisor. Using this security architecture can lead us to better performance by avoiding unnecessary traffic and dedicating the cloud resources to necessary traffics. In the next chapter we will implement the prototype of our security architecture using the tools that we explained in this chapter. After that we can observe the results to see if the proposed architecture can improve the integrity and confidentiality of information in the multi-tenancy cloud environments.

CHAPTER 4 VALIDATION AND PERFORMANCE ANALYSIS

In the previous chapter, we presented our security architecture in detailed that intends to meet the system requirements. Now, in this chapter we validate this work in order to guarantee that it certainly achieves the requirements. Early in the first section we define mathematical equations for validating our security architecture. Then we present the numerical results and the last section will be dedicated to analyzing the results.

4.1 Mathematical equations

In this section, we define mathematical equations to evaluate our proposed security architecture that we called it CloudGuard. First of all, we analyze the system throughput of our architecture as compared to another security architecture that we have studied in the literature review. While different solutions can be found in the literature we chose CloudPolice[43] to compare it to our security design because as explained in chapter two, it is a hypervisor-based access control system and like our security designed it implements access control only within the hypervisor. In other words, both of these security architectures (CloudPolice and CloudGuard) are designed to enforce the access control list in the hypervisor. Secondly, we analyze the latency of CloudGaurd as compared to CoudPolice.

CloudPolice vs. CloudGuard

A major different between CloudPolice and CloudGuard is that in our security design, the source hypervisor waits for control packet response from destination hypervisor and if the security policies of the source VM and destination VM match, then the source hypervisor passes the traffic toward the destination hypervisor and consequently destination hypervisor passes the traffic to destination VM.

Contrary, according to CloudPolice, the source hypervisor does not wait for control packet response and it sends the whole traffic to destination hypervisor. When the traffic reaches the destination hypervisor, then the security policies will be checked between the traffic and destination VM. In this case, if the traffic is authorized, then it can continue its journey to destination VM. But, if the traffic does not match with the security attributes of destination VM, the traffic will be ignored and dropped in the destination hypervisor.

4.1.1 Parameters

In order to form a mathematical equation to evaluate the CloudGuard, we considered some parameters. These parameters are driven from the important element of a virtualized network system, such as link bandwidth, size of the packet and etc. So, we defined the following parameters to achieve our objectives:

Table 4.1 List of parameters used in equations

N_{pkt}	Number of packets transmitted from source VM to destination VM;
S_{pkt}	Size of the data in each packet (bits);
N_{flow}	Number of traffic flows that are being transmitted from source VM to destination VM;
S_{flow}	Size of the traffic flow (bits);
$Time$	The amount of time it takes for a traffic flow to be sent from source VM to destination VM (second);
RTT	The amount of time it takes for a control packet to be sent from source hypervisor to destination hypervisor and received back again by source hypervisor (second);
BW	The amount of data that can be carried from one point to another in a given timer period in a gigabit Ethernet (1.073.741.824 bits per second);
P_{int}	Probability of intrusion traffic

Flow size:

Calculating the size of the traffic flow is the first step in forming a mathematical equation. Request for Comments (RFC) is a publication of the Internet Engineering Task Force (IETF). In march 2004 this publication has published the proposal RFC-3697 [50] and defines the traffic flow as "an sequence of packet sent from a particular source to a particular unicast, anycast, or multicast destination that the source desires to label as a flow". Based on this definition we can formulate the flow size as: Size of flow = Size of packet × Number of packets

In the above calculation, we replaced the packet size and number of packets with the "sequence of the packet" that we had in the RFC-3697 definition.

Also we considered the size of each packet equal to 128 bytes that is equal to 1024 bits. Besides that, we considered 1000 packets in each traffic flow and finally we assume that there are 1000 to 10000 traffic flows that are going to pass from source VM to the destination VM. So, the size of the flow in our experiment would be 1024000 bits.

Probability of intrusion traffic:

Intrusion traffic in our case is the amount of traffic that is generated by a malicious VM and the target of this traffic is to penetrate vulnerabilities of destination VM. The amount of intrusion traffic is not predictable in a multi-tenancy cloud network. This amount may vary from zero to hundred percent depending on the value of the assets that are in the cloud network.

Round trip time (RTT):

Because the process of generating control packet is very similar to the process of generating control packets in CloudPolice, we assume that the RTT of control packet is the same as CloudPolice. In fact CloudPolice states that it adds ≈1.5ms round-trip latency [49], accordingly we assume the same amount of RTT for control packet in CloudGuard.

4.1.2 Latency

Latency is one of the parameters that we consider in our project. Having all the parameters in mind now we can generate two equations for calculating the latency of our security architecture as well as CloudPolice. By latency, we mean the time interval between stimulation and response. In fact we want to measure which security architecture is able to transmit the data with less delay and overhead.

CloudGuard:

In our system, stimulation is generating the control packet by source hypervisor and the response is receiving the *control packet response* back by the source hypervisor. The control packet response is generated by the destination hypervisor. When we wanted to explain the RTT parameter (section 4.1.1), we mentioned ≈1.5ms round-trip latency for control packet and we assumed that the RTT for control packet in our security architecture is the same as CloudPolice. Therefore, by knowing the control packet RTT delay, we want to calculate the delay time for sending certain amount of packet from source VM to the destination VM.

Looking back again into Figure 3-12 Process sequence diagram, we see that no matter what kind of traffic (authorized or not-authorized) is passing through the hypervisor, CluodGuard generates the control packet. When one VM wants to communicate with another VM in the same cloud environment, the source VM first sends the traffic to the source hypervisor. Then the source hypervisor checks the destination address and find out the destination hypervisor on which the destination VM is hosted on. In the next step, the source hypervisor generates a control packet and send this control packet to the destination hypervisor. The source hypervisor keeps the traffic and do not let the traffic to be passed to the destination hypervisor unless it receives back the control packet response from the destination hypervisor. The responsibility of the destination hypervisor in this stage is to check the security attributes of the traffic with the security policies of the destination VM. If the traffic is authorized, the control packet response sends specific signal to the source hypervisor and tells it to pass the traffic.

Therefore, for every single traffic flow, CloudGuard generates one control packet. Based on this fact, to find out average latency of CloudGuard we just need to multiply the number of traffic flow to the round trip time which is ≈1.5ms. To derive an equation for our evaluation, first we add RTT and the time it takes to transmit one traffic flow. Then we multiply it by probability

of intrusion. After all we should add this value to the round trip time of the control packet for intrusion traffic. Finally, we repeat this process for each traffic flow in order to have the average latency of CloudGuard security architecture.

$$C_{Guard}\ Average\ Latency = [(RTT + \frac{S_{flow}}{BW}) * (1 - P_{int}) + (RTT * P_{int})] * N_{flow} \quad (4.1)$$

Where: $0 \leq P_{int} \leq 1$; $RTT \approx 1.5ms$; $S_{flow} = (S_{pkt} * N_{pkt})$.

It is obvious that presence or lack of intrusion traffic does not change the mechanism of this security architecture and CloudGuard generates the control packet in any conditions. Equation 4.1 is based on the number of traffic flows. Because for each traffic flow we have a control packet and the round trip time for that control packet is not all, in fact, based on the size of traffic flow, it takes certain amount of time to transmit the traffic flow from source to destination. That is why we have $(RTT + \frac{S_{flow}}{BW})$ in the first part of our equation.

Then, the amount of time that we calculated in the first part of the equation should be multiplied by the probability of intrusion traffic $(1 - P_{int})$ because certain percentage of traffic flow is unwanted and intrusion traffic.

Now we have the time that we calculated in the previous part, this time should be added to the round trip time that intrusion traffic impose to the system $(RTT * P_{int})$. The fact is that, the intrusion traffic is not being sent to the destination but there will be a little delay only for the control packet which is being sent from source hypervisor to the destination hypervisor.

Finally, the whole process needs to be multiplied by the number of traffic flow because for each traffic flow we have a control packet.

As explained before, the probability of intrusion traffic is always between zero and one. Also, based on CloudPolice, we suppose that RTT is $\approx 1.5ms$.

CloudPolice:

At the beginning we look back again into mechanism of CloudPolice and according to that mechanism we define an equation for this security architecture. When a VM wants to communicate with another VM in the same cloud environment, the source VM sends the traffic to the source hypervisor. The source hypervisor checks the destination VM address and pass the traffic to the proper destination hypervisor on which the destination VM is hosted. Then the destination hypervisor check the security attribute of the traffic with the security policies of the destination VM. If the traffic is not authorized, the destination hypervisor generates a control packet and sends this control packet to the source hypervisor to tell the source hypervisor to block the next stream of the same traffic or to limit the bandwidth of the network that is allocated to this traffic.

Therefore, CloudPolice generates control packet only for traffic flows that are intrusion. So, the number of control packets that are generated by CloudPolice is depended on the percentage of intrusion traffic. Thus, for calculating the total latency of CloudPolice we only need to multiply the number of intrusion traffic by the RTT. Then we need to calculate the time that takes for intrusion traffic to pass to the network because this amount of intrusion traffic can technically consume the available bandwidth of the network.

Based on the above explanation, for calculating average latency of CloudPolice first we calculate the round trip time of all control packets (because CloudPolice generates the control packet only for intrusion traffic) and then we add this to the time that it takes to pass the intrusion traffic from source to the destination.

$$C_{Police}\, Average\ Latency = [(RTT * P_{int}) + \left(\frac{S_{flow}}{BW}\ (1 - P_{int})\right)] * N_{flow} \quad (4.2)$$

Where: $0 \leq P_{int} \leq 1$; RTT$\approx 1.5ms$; $S_{flow} = (S_{pkt} * N_{pkt})$.

As you can see in the equation, the probability of intrusion traffic plays a great role in the result of the equation because CloudPolice generates the control packet only for intrusion traffic.

This equation is also based on the number of traffic flows. Therefore, we should multiply the round trip time of the control packets by the probability of intrusion traffic $(RTT * P_{int})$ in order to have the delay of intrusion traffics.

Then, this delay time needs to be added by the time that it takes to pass the traffic flow from source VM to the destination VM. And as you see in the equation 4.2, we multiplied it with the intrusion traffic $\left(\frac{S_{flow}}{BW}\ (1 - P_{int})\right)$.

Finally, we repeat this calculation for each traffic flow in order to calculate the average delay of CloudPolice.

Like the equation 4.1, the probability of intrusion traffic is always between zero and one. Also, based on CloudPolice, we suppose that RTT is $\approx 1.5ms$.

4.1.3 System throughput

System throughput is another item that we are going to take into considerations in order to evaluate our security architecture. We defined two equations, one equation for CloudPolice and another equation for CloudGuard.

CloudGuard:

First we derive an equation for calculating system throughput of CloudGuard. Figure 3-12 Process sequence diagram can help us to derive an equation. As explained before, CloudGuard generates a control packet for every traffic flow before passing the traffic to the destination hypervisor. The traffic remains in the source hypervisor until the source hypervisor received back the control packet response. So, unlike CloudPolice, only authorized traffic can pass through the network if CloudGuard is used in a multi-tenancy cloud system.

For calculating the system throughput of CloudGuard we need to divide number of traffic flow by the average latency of the CloudGuard. Then we multiply that with the intrusion traffic.

$$C_{Guard}\ System\ Throughput = \frac{(S_{flow} * N_{flow})}{Latency\ C_{Guard}}\ (1 - P_{int}) \tag{4.3}$$

Where: $0 \leq\ P_{int} \leq 1$; $RTT \approx 1.5ms$; $S_{flow} = (S_{pkt} * N_{pkt})$.

Unlike equation 4.1 and 4.2, this equation is based on probability of intrusion traffic. All we did in this equation is to divide the total size of traffic flow by the average latency that we

calculated in equation 4.1. The reason for this calculation ($\frac{(S_{flow*}\, N_{flow})}{Latency\ C_{Guard}}$) is to find out the amount of traffic flow that we can pass from source VM to the destination VM in a certain time. Finally we multiply the first part of equation 4.3 by the probability of intrusion traffic.

CloudPolice:

If we look deeper into this security architecture, we can see that it does not send any control packet before passing the traffic packets to the destination hypervisor. In other words, CloudPolice pass all traffics from source hypervisor to destination hypervisor, and it is the destination hypervisor that decide about the destiny of the traffic.

Unlike our security architecture, CloudPolice let all traffic flows to pass from one hypervisor to another hypervisor that includes unwanted and unauthorized traffic. Based on this fact, when we want to calculate the system throughput of CloudPolice we should deduct the amount of unauthorized traffic. The reason for this deduction is that the unauthorized traffic that is generated by an intruder can occupy the usable bandwidth of network.

So, like equation (3), we need to divide number of traffic flow by the average latency of the CloudPolice. Then we multiply that with the intrusion traffic. This equation is for the situation in which the probability of intrusion traffic is between zero percent up to fifty percent, but because of the nature of CloudPolice we should consider the system throughput of CloudPolice equal to zero if the probability of intrusion is between fifty percent up to hundred percent.

$$C_{Police}\, System\ Throughput = \frac{(S_{flow*}\, N_{flow})}{Latency\ C_{Police}}\ (1 - 2P_{int})\ \text{, If} \qquad 0 \leq\ P_{int} \leq 0.5 \tag{4.4}$$

OR

$$C_{Police}\, System\ Throughput =\ 0\ \text{, If} \quad 0.5\ \leq\ P_{int} \leq 1$$

As you see, the equation for calculating the system throughput of CloudPolice has a conditional situation. This condition depends on amount of intrusion traffic. In other words, if the probability of intrusion traffic is between zero and fifty percent, we can calculate the system throughput of CloudGuard and CloudGuard is the same way and with only a slight change for

adjusting the probability of intrusion. But if the probability of intrusion is more that fifty percent, the system throughput of CloudPolice will be equal to zero.

4.2 Numerical results

In this section we are going to we give value to variables of equations and take some numerical results from the equations. P_{int} is a variable that we give different values to that to run our test. The values that we give to P_{int} are just random numbers chosen from 0 and 1 (0% to 100%). The other variable is the number of traffic flows that we choose again random numbers from 1000 to 10000. The reason for choosing those numbers is to verify how our security architecture performs in different situations. Then we will analyze the results.

4.2.1 Latency

Latency or in cloud computing system is an important element that can indicate the effectiveness of a security system. Before getting into the tests and results, we should mention a vital point about all security system. That important point is that, every single security point imposes some delay and latency to the whole system. Not only in the computing systems, but also in the not computing systems we have the same fact. For example, in an airport if we want to increase the level of security of the passenger we need to place a security gate and force people to pass through this security gate. The guardians need to spend few minutes for investigating all people entering the airport. Everybody agree that the delay due to security check in the airport is absolutely logic and acceptable. We can generalize that fact to our security architecture. And we can summarize that the delay of our security system has an acceptable logic.

To evaluate the latency of our security architecture we perform nine different tests as illustrated in Table 4.2. In the first test, 10 percent of all traffics are intrusion. In the second test, 20 percent of the traffics are intrusion and we continue this strategy till test number nine in which 90 percent of traffics are intrusion. In all tests, we have a fix round trip time ≈1.5ms, which is the RTT, proved by CloudPolice. Again we emphasize that; we want to calculate the delay time for

sending certain amount of packet from source VM to the destination VM. In fact we want to measure which security architecture as able to transmit the data with less delay and less overhead.

Table 4.2 Latency variables and values.

Test Number	P-Intrusion
1	0.1
2	0.2
3	0.3
4	0.4
5	0.5
6	0.6
7	0.7
8	0.8
9	0.9

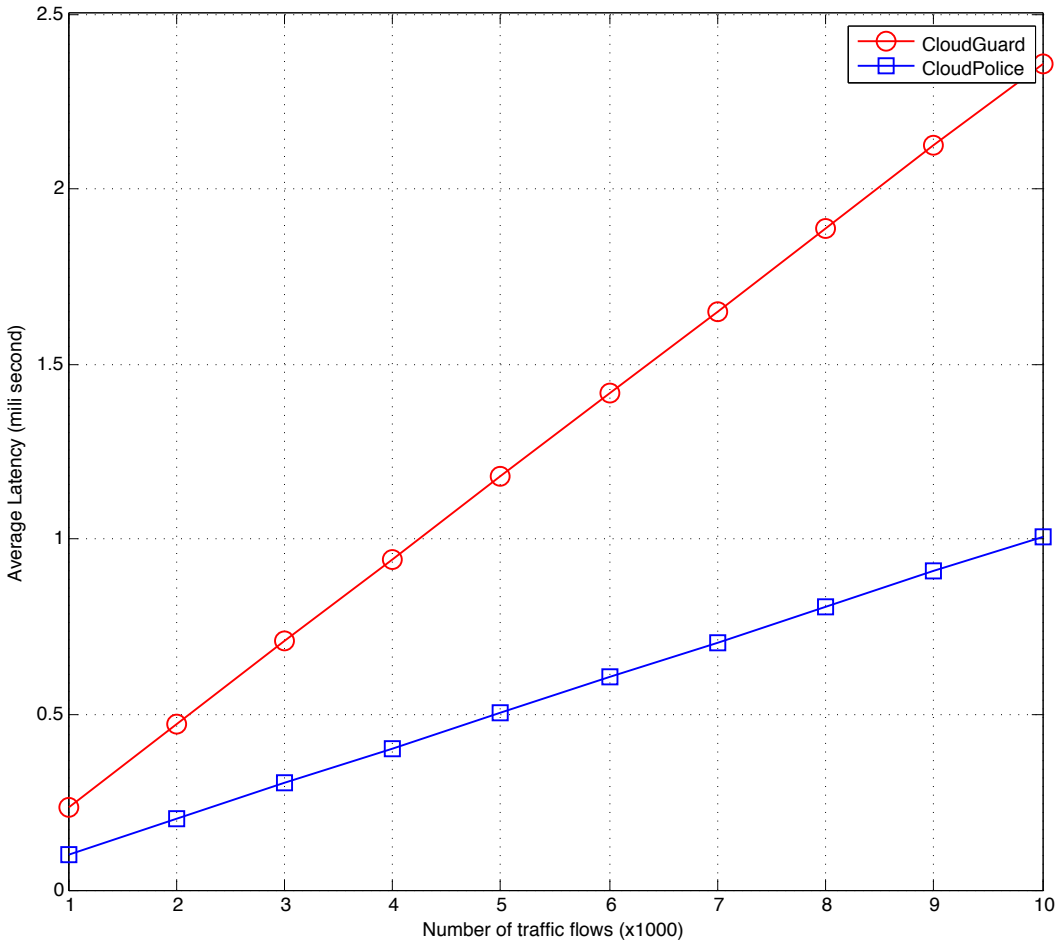

Figure 4-1 Test number 1 (P-int = 0.1)

Test number 1

Test number one takes place in a cloud environment in which 10 percent of its traffic is unauthorized and intrusion. In comparison with CloudPolice, our security architecture has more latency and overhead on the system. When we start the test by 1000 traffic flows, the latency that CloudGuard is more than CloudPolice but it is very close to that. Then the delay becomes more when the numbers of traffic flows grow. We explained in the previous chapter that, CloudPolice generates the control packet only if there is any intrusion traffic in the system. That is why the latency of CloudPolice is close to CloudGuard when there is small amount of intrusion traffic in the system.

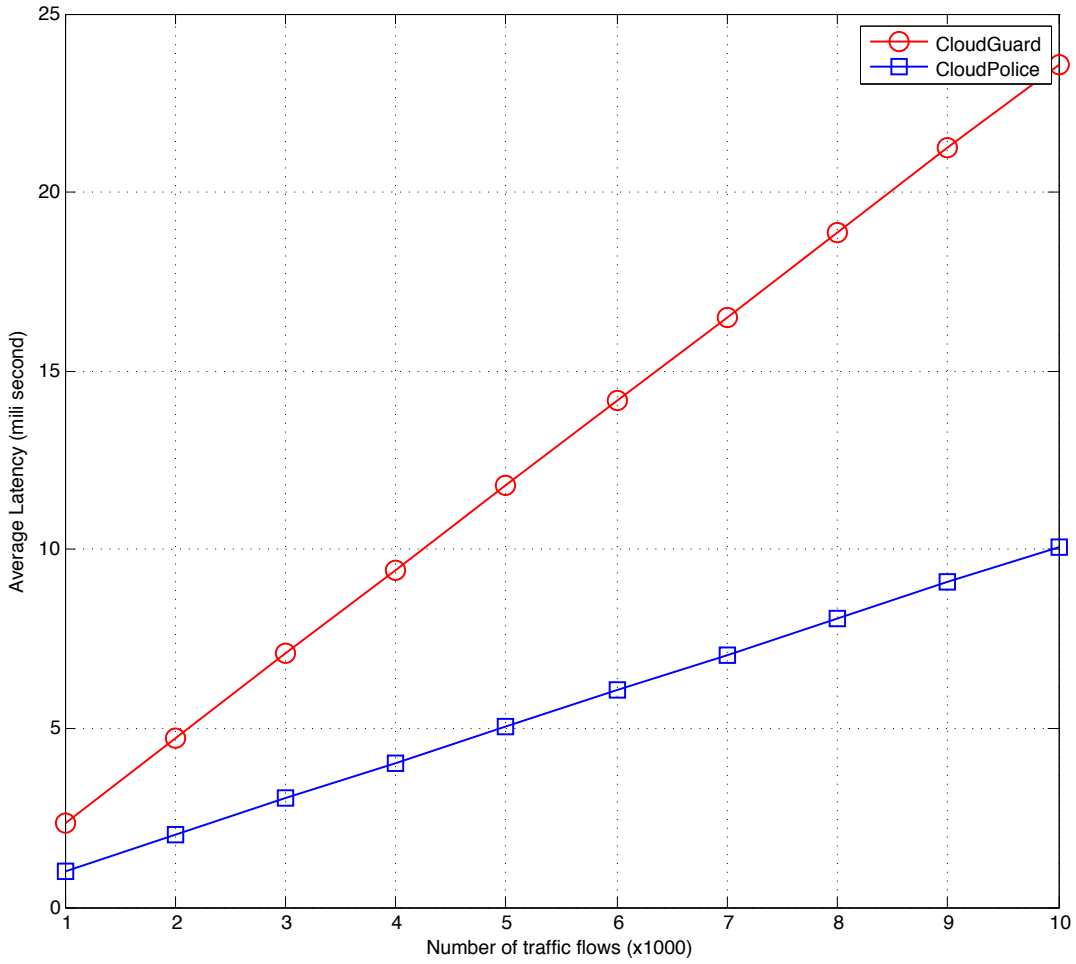

Figure 4-2 Test number 2 (P-int = 0.2)

Test number 2

For the test number two we assigned the P-int variable to 0.2. This value means that 20 percent of the traffic that is in the cloud environment is intrusion and the rest of traffic flows are healthy traffic. In contrast with the test number one, we see that the difference between average latency of CloudGuard and CloudPolice is getting closer. The more is the probability of intrusion traffic in the system, the lower become the latency of CloudGuard. In the next test we will increase the probability of intrusion traffic and we will investigate the results.

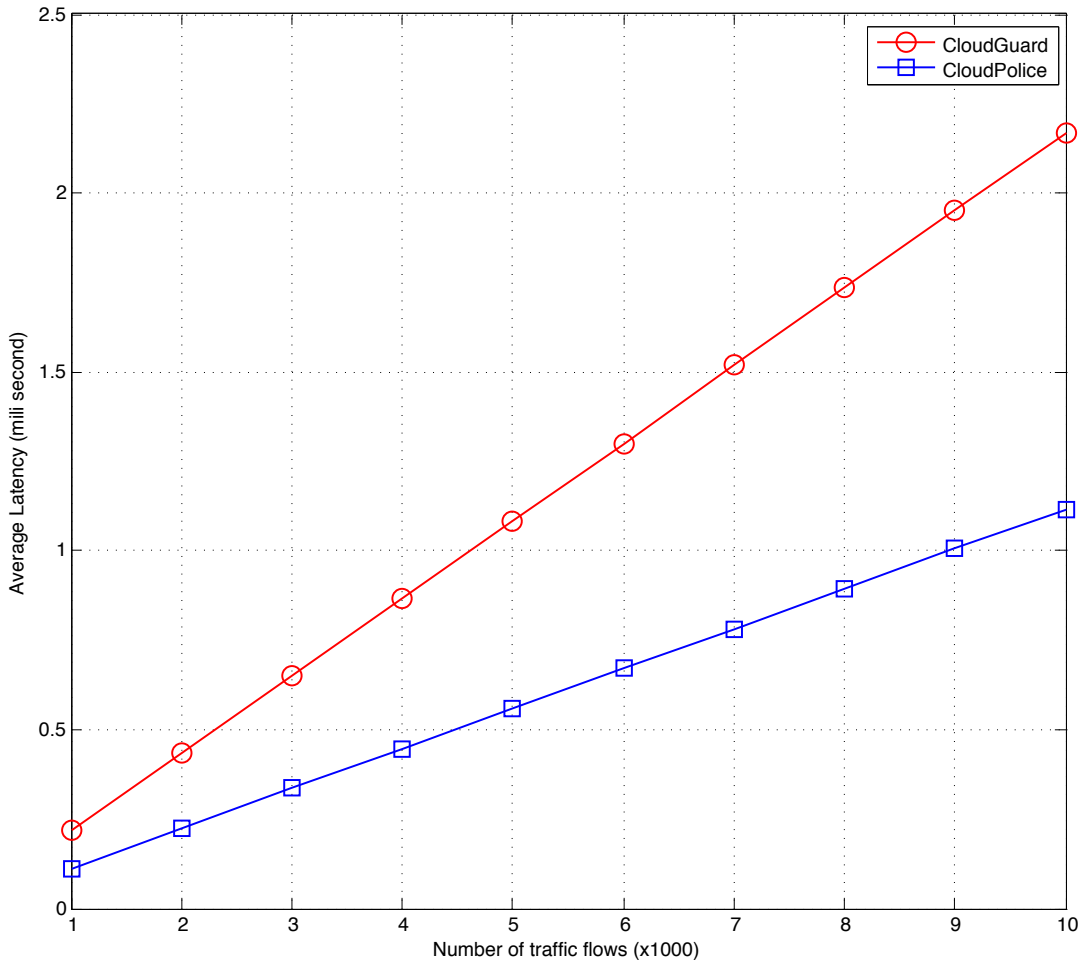

Figure 4-3 Test number 3 (P-int = 0.3)

Test number 3

For the test number 3, we set the probability of intrusion traffic to 0.3. In this test we assume that 30 percent of the traffic on the cloud system is malicious and intrusion traffic. Figure 4-3 reveals that latency of CloudGuard is getting much better than the latency of CloudPolice when the probability of intrusion traffic is getting higher in a multi-tenancy cloud environment.

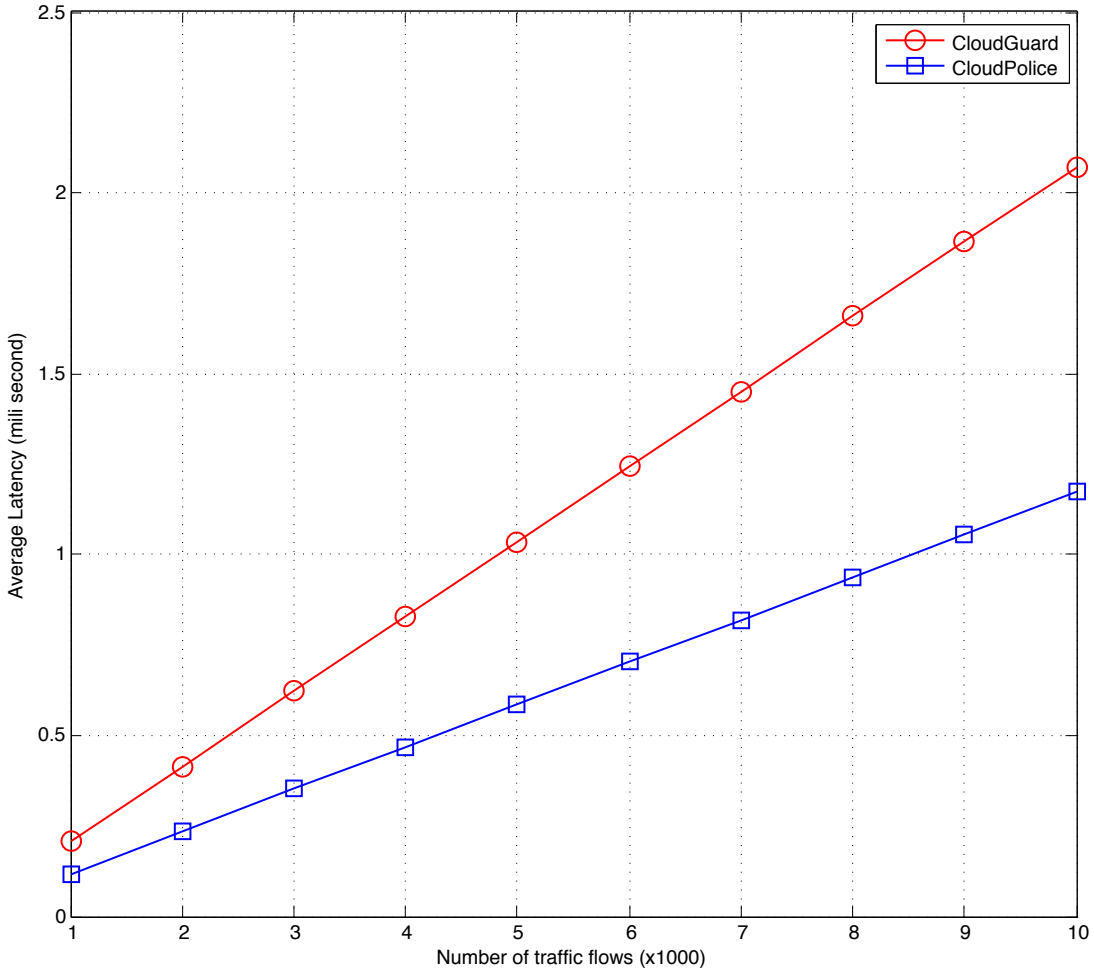

Figure 4-4 Test number 4 (P-int = 0.4)

Test number 4

For the test number 4, we set the probability of intrusion traffic to 0.4. In this test we assume that 40 percent of the traffic on the cloud system is malicious and intrusion traffic and still we observe that the average latency of CloudPolice is less than the average latency of CloudGuard. So, in this test the performance of CloudGuard is getting closer to the performance of CloudPolice in the context of average latency.

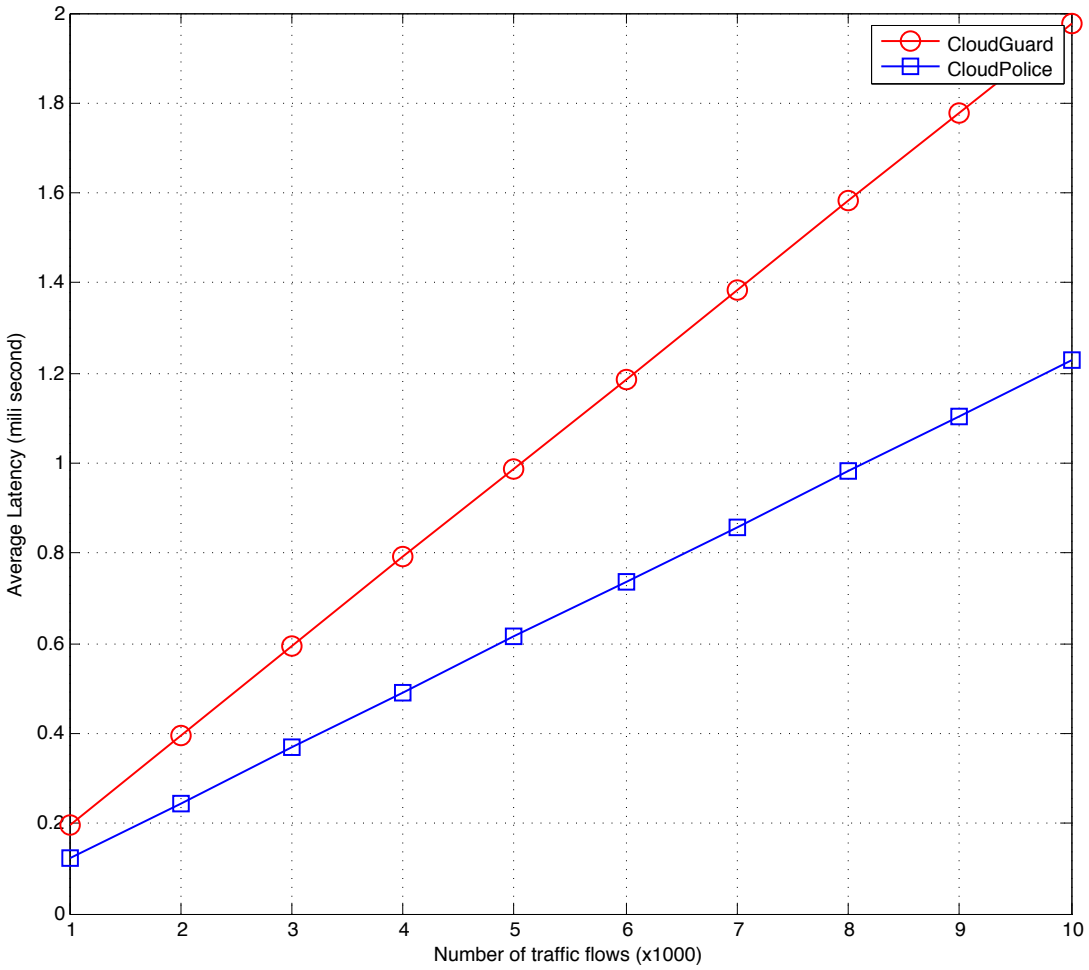

Figure 4-5 Test number 5 (P-int = 0.5)

Test number 5

We continue our experiments and reach to the test number five. We gave the value of 0.5 to P-int. In this test we assume that 50 percent of the traffic on the cloud system is malicious and intrusion traffic. Figure 4-5 indicates that the average latency of CloudGuard is still more than the average latency of CloudPolice but the difference between average latency of CloudGuard and CloudPolice is decreasing.

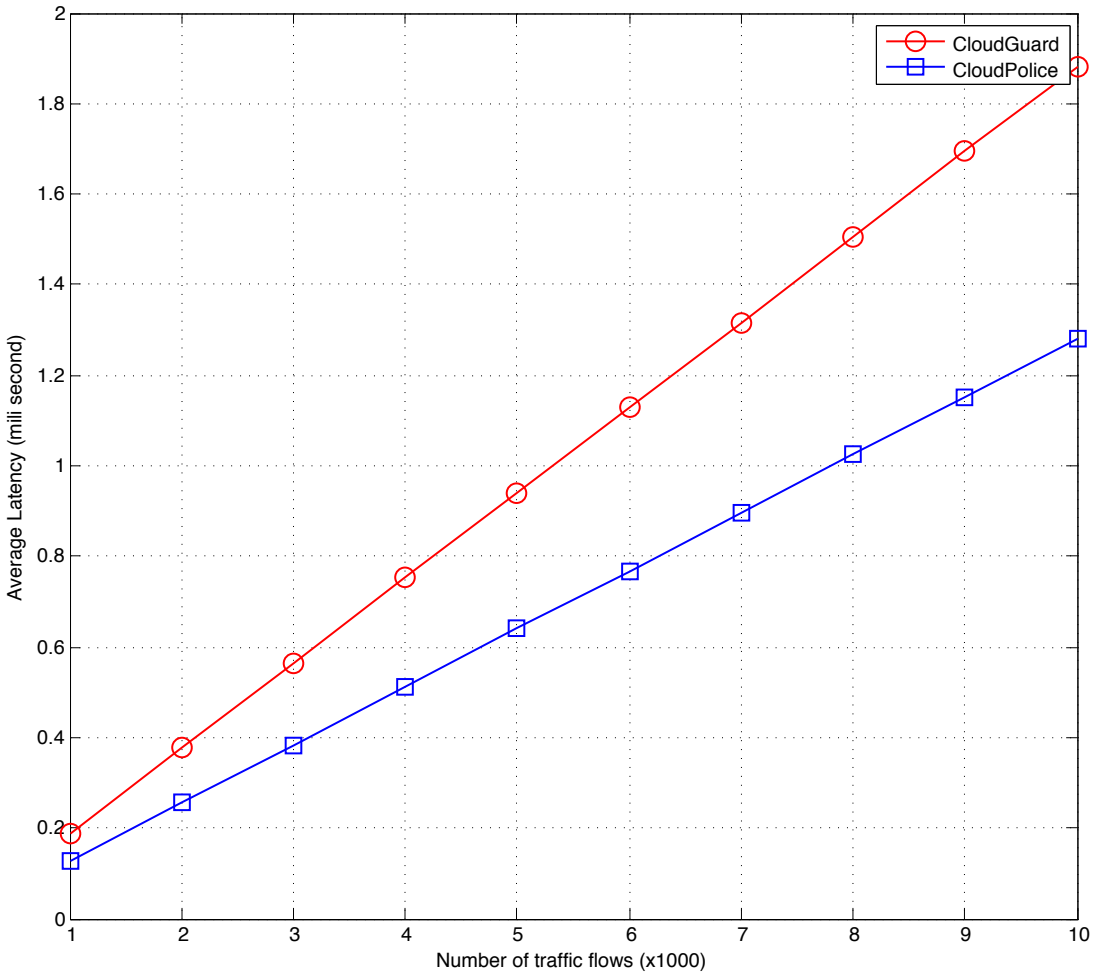

Figure 4-6 Test number 6 (P-int = 0.6)

Test number 6

The probability of intrusion traffic for the test number six is sixty percent. As illustrated in the result of the test number 6, if we compare the average latency we can see that our security architecture impose only a little bit of overhead on the system in contrast to that of CloudPolice. It is important to mention that, the more we increase the probability of intrusion; the average latency that CloudPolice imposes to the system is getting more.

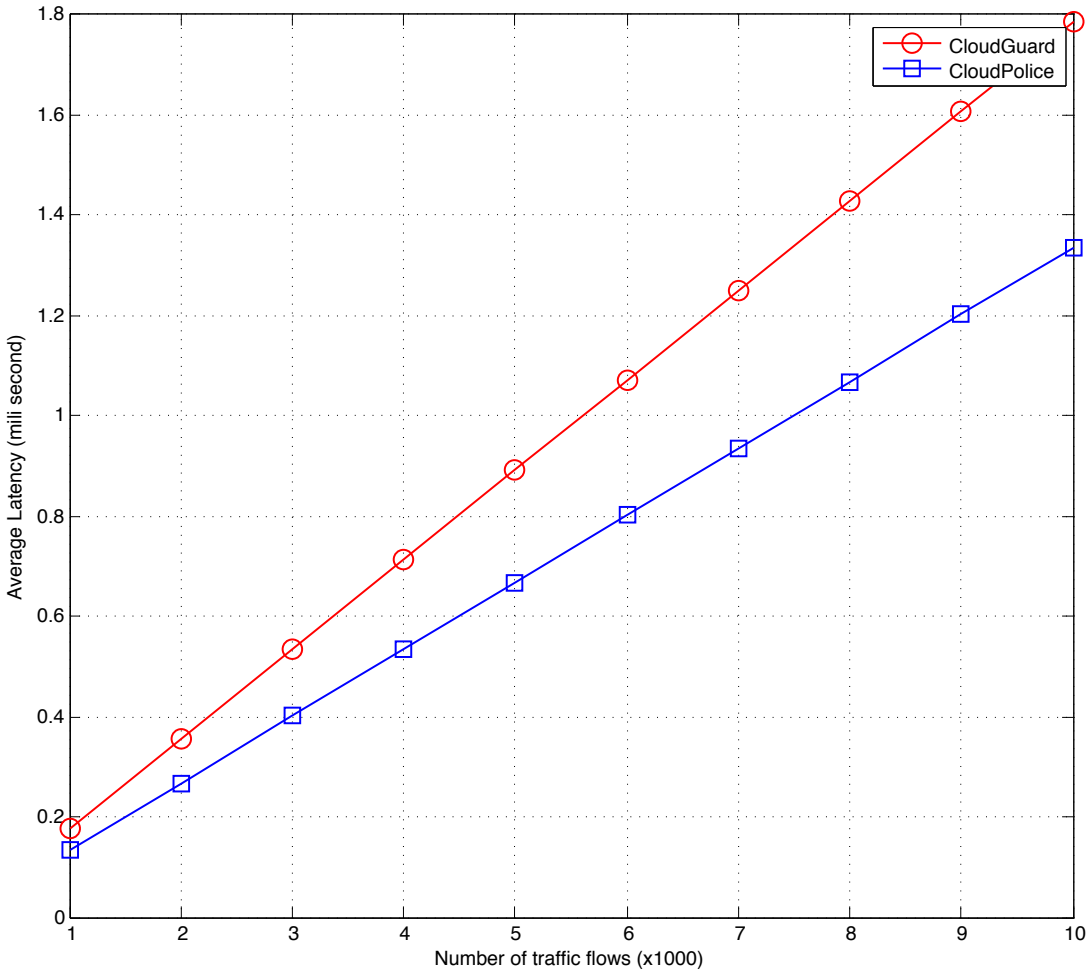

Figure 4-7 Test number 7 (P-int = 0.7)

Test number 7

The test number 7 is the completion of previous tests. The goal of this test is again to increase the probability of intrusion traffic to seventy percent. Although CloudPolice is generating control packet for seventy percent of traffic flow but the average latency of this security architecture is getting very close to the average latency of CloudGuard. Figure 4-6 reveals that latency of CloudGuard is getting much better than the latency of CloudPolice when the probability of intrusion traffic is getting higher in a multi-tenancy cloud environment.

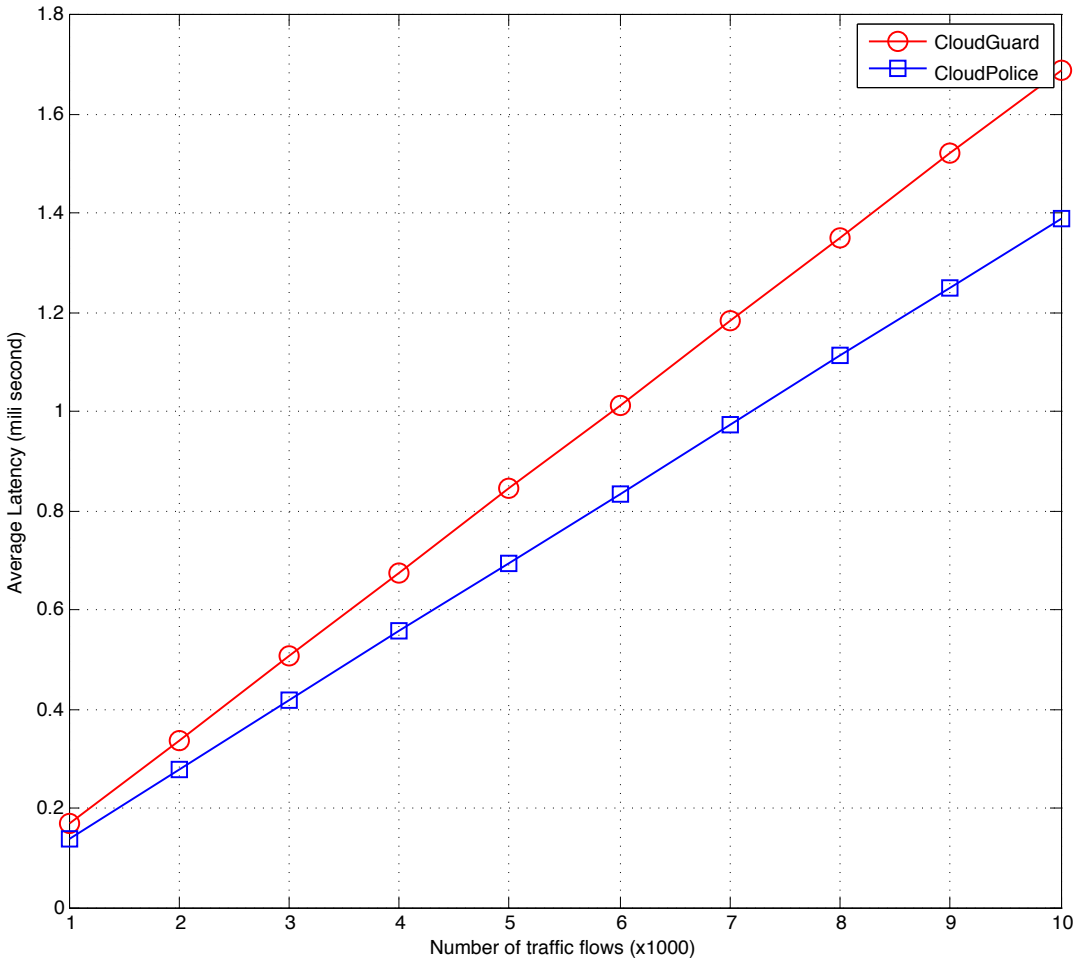

Figure 4-8 Test number 8 (P-int = 0.8)

Test number 8

We continue our experiments with the test number 8 and assign the value of 0.8 to the P-int variable. The goal of this test is again to increase the probability of intrusion traffic to eighty percent. Although CloudPolice is generating control packet for eighty five percent of traffic flow but the latency of this security architecture is way more than the average latency of CloudGuard. Figure 4-6 reveals that latency of CloudGuard is getting much better than the latency of CloudPolice when the probability of intrusion traffic is getting higher in a multi-tenancy cloud environment.

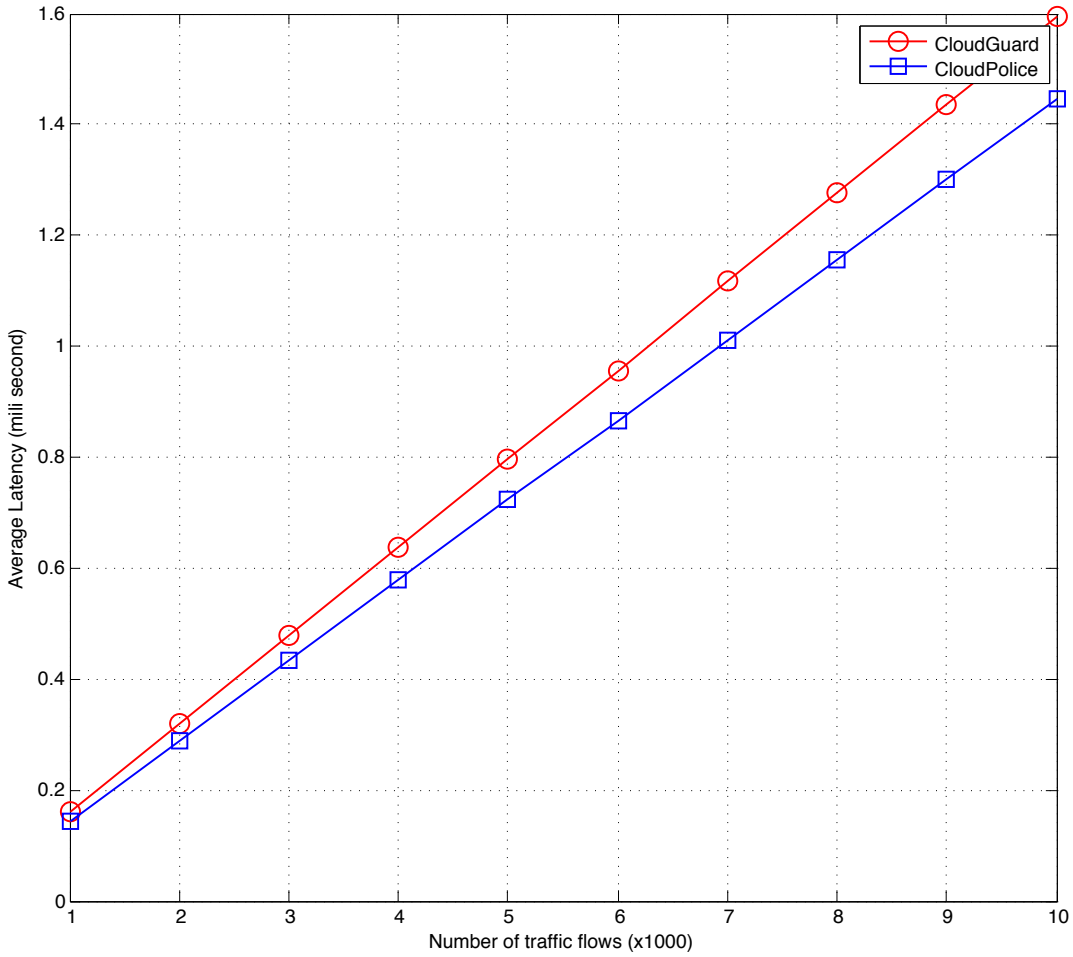

Figure 4-9 Test number 9 (P-int = 0.9)

Test number 9

This test is the last one for evaluating the latency. We increased the value of probability of intrusion traffic to 90 percent (0.9). We observed that the average latency of CloudPolice and CloudGuard are almost the same. This test reveals that latency of CloudGuard is almost the same as the latency of CloudPolice when the probability of intrusion traffic is getting higher in a multi-tenancy cloud environment.

The above tests show that CloudGuard imposes a little bit of overhead in the system. This latency is less than a millisecond. Thus, in a high-risk cloud environment that number of intrusion traffic is high, the latency of CloudGuard is very close to the latency of CloudPolice.

4.2.2 System throughput

To illustrate the system throughput of our security architecture we perform only 1 test. In this test the number of traffic flow (N_{flow}) is fix and equal to 1000. But for the other variable that is probability of intrusion traffic (P_{int}) we start our test from 0 percent of intrusion traffic to 100 percent of intrusion traffic.

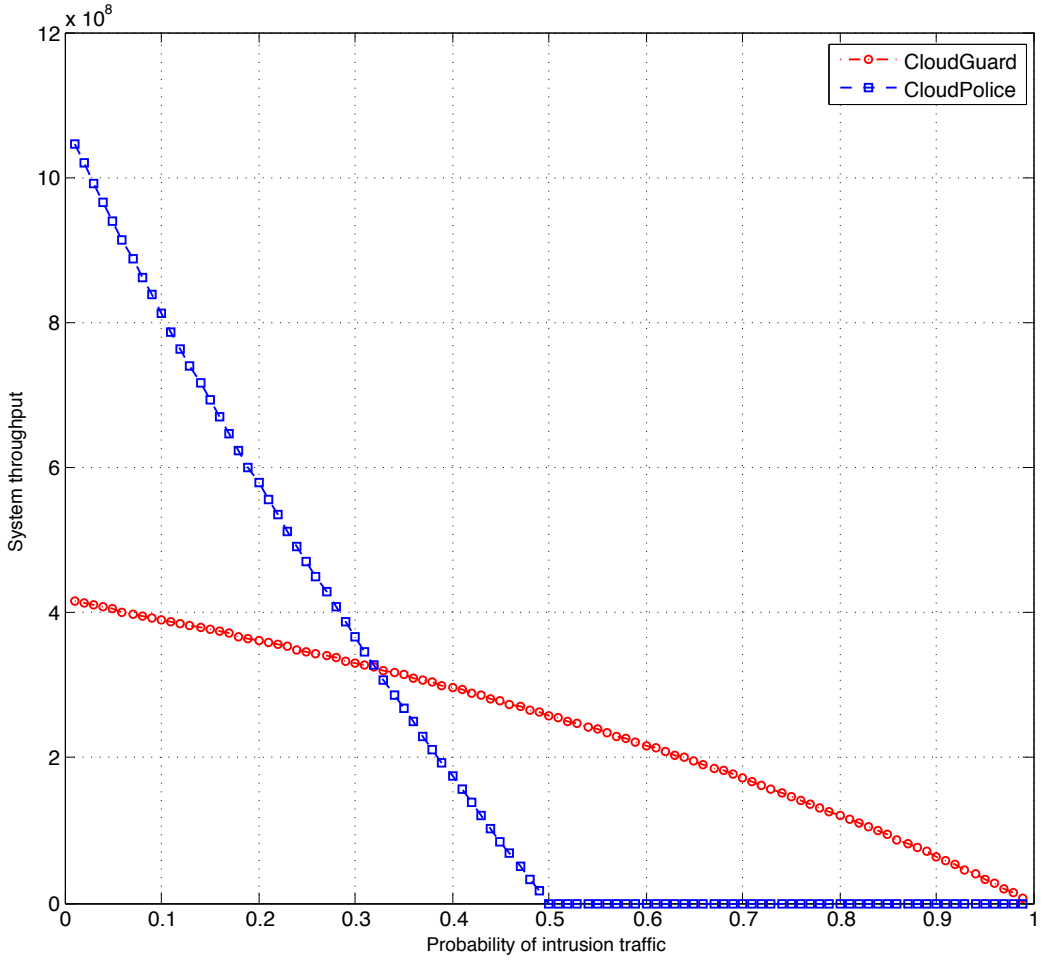

Figure 4-10 Test number 10 for evaluating system throughput

As illustrated in the Figure 4-10, we start the test from probability of intrusion of 0, the system throughput of CloudPolice is better, but when reach to the point that probability of intrusion is about thirty percent (0.3215) both CloudPolice and CloudGuard has the same throughput and we continue to the end of the test, our security architecture show better system throughput. Important point in this test is that after 50 percent of intrusion traffic, CloudPolice is paralyzed and has not system throughput.

Result of the pervious test indicates that CloudGuard has better system throughput than CloudPolice in high-risk multi-tenancy cloud computing environments. Therefore, we can conclude that CloudGuard is effective security architecture and is suitable for high-risk environments. High-risk multi-tenancy cloud computing environment is a cloud environment in which, the system is target of many attacks and there are lots of intrusion and unwanted traffic being sent to VMs. CloudGuard has obviously better system throughput in the critical multi-tenancy cloud network environments. The results show that our security architecture is more effective when the amount of intrusion traffic is higher in the cloud system. Financial institutions and game centers are examples of network that are always target of attackers and malicious agents. At the same time, we can hardly imagine a network environment with zero percent of attacks.

CHAPTER 5 CONCLUSION

In this dissertation we have proposed an architecture that controls the access of network entities in a multi-tenancy cloud environment. This architecture implements a hypervisor-based access control mechanism and allows only legitimate and authorized traffic to pass to the cloud network environment.

This last chapter presents a summary of the work that has been exposed in this project. Later on, future works to overcome the limitation of the current proposal are described.

5.1 Summary of the work

We propose CloudGuard, a system that implements a hypervisor-based access control mechanism. Based on the concept of *security-in-depth*, CloudGuard adds another layer of security to multi-tenancy cloud computing environments and prevents unauthorized and illegal access to the cloud resources. This security architecture can be simply implemented to hypervisor and provide scalable and more robust access control than existing network-based techniques.

When the source VM sends the traffic to destination VM, the traffic has to pass through the hypervisor that it hosting source VM. This source hypervisor recognizes the destination IP address of the traffic and sends the traffic to the destination VM. If the destination VM is located on the same hypervisor, the traffic will be simply delivered to the destination. But if the destination VM is located on another hypervisor, then, it generates a control packet. This control packet consists of necessary information for access control checking as well as source IP address of the packets, destination IP address of the packets, port number and protocol type (TCP or UDP). So the control packet has to be sent to destination hypervisor. Destination hypervisor checks the content of the control packet and decide whether the traffic can be delivered to destination hypervisor or not.

If the source VM is permitted to send the so-called traffic to the destination VM, so the destination hypervisor adds the pass value to the control packet payload and sends it back to the source hypervisor. But if the source VM is not permitted to send that traffic to the destination VM, then the destination hypervisor adds the drop value to the control packet payload and sends it back to the source hypervisor. Based on the pass or drop values, the source hypervisor threat

the awaiting traffic. This access control architecture allows us to keep the bandwidth of the network in a healthy level.

We further built a mathematical model in order to evaluate the performance of our security architecture. Using that mathematical model, we conducted few tests to evaluate the system throughput and the latency of our security architecture.

Although our security architecture impose small amount of delay (less than one millisecond) to the system, but the results show that the system throughput of our access control security architecture is better in high-risk environments where the number of intrusion traffic is high. The more the amount of intrusion traffic is, the better the system throughput of our security architecture becomes in contrast with CloudPolice. For example, when the probability of intrusion traffic reaches 32 percent of whole traffic, CloudPolice becomes almost paralyzed but our security architecture can still pass the authorized traffic from source VM to the destination VM.

5.2 Limitations

As illustrated in the previous chapter, we evaluated our security architecture by using mathematical modeling. The test bed of this evaluation is included a set of virtual machines that are located in the hypervisors. This virtualized environment causes limited scope of the experiments we could preform. For example, we were not able to evaluate the latency and throughput of our security architecture when traffic is being sent between a non-VM nod and a virtual machine. However, this limitation in our evaluation has no bearing on how our security architecture can reduce unwanted and unauthorized traffic in multi-tenancy cloud computing environments.

5.3 Future works

Future work should focus on the intra-cloud traffic. In a real world cloud environment, the traffic may pass through lots of middle devices such as switches, routers and firewalls. So, when a VM wants to send traffic to another VM that is located in another cloud, we need more complex access control update between hypervisors that are located in different cloud. Hence, in the context of intra-cloud traffic, the process of propagation of the security attributes and updates between hypervisors should be taken into considerations. An example of security update is when

one virtual machine migrates from one hypervisor to another hypervisor, so other hypervisors should be updated about this change in with acceptable delay.

BIBLIOGRAPHY

[1] D. Catteddu and G. Hogben, "Report on Cloud Computing Security Risk Assessment , Benefits, Risks and Recommendations for Information Security," European Network and Informaiton Security Agency (ENISA), Athens, GreeceNovember 2009.

[2] K. Hashizume, D. G. Rosado, E. Fernández-Medina, and E. B. Fernandez, "An analysis of security issues for cloud computing," Journal of Internet Services and Applications 4:5, vol. 4, 2013, pp. 5-18.

[3] Z. Minqi, Z. Rong, X. Wei, Q. Weining, and Z. Aoying, "Security and Privacy in Cloud Computing: A Survey," in *6th International Conference Semantics Knowledge and Grid (SKG),* , Beijing, 2010, pp. 105-112.

[4] C. J. Guo, W. Sun, Y. Huang, Z. H. Wang, and B. Gao, "A framework for native multi-tenancy application development and management," in *9th International Conference on E-Commerce Technology/4th International Conference on Enterprise Computing, E-Commerce and E-Services*, Tokyo, 2007, pp. 551-558.

[5] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud Computing and Grid Computing 360-Degree Compared," in *GCE '08 Grid Computing Environments Workshop,* , Austin, TX, 2008, pp. 1-10.

[6] Z. Wang, "Security and Privacy Issues within the Cloud Computing," in *International Conference on Computational and Information Sciences (ICCIS),* , Chengdu, China, 2011, pp. 175-178.

[7] N. I. o. S. a. Technology, "The NIST Definition of Cloud Computing," SP 800-145 ed: National Institute of Standards and Technology, 2011, p. 7.

[8] N. I. o. S. a. Technology, "NIST Cloud Computing Reference Architecture," vol. NIST Special Publication 500 - 292, ed: National Institute of Standards and Technology, 2011, p. 35

[9] IBM. (2012, 5 Feb 2013). *What is cloud computing?* Available: http://www.ibm.com/cloud-computing/ca/en/what-is-cloud-computing.html

[10] W. Liu, "Research on cloud computing security problem and strategy," in *2nd International Conference on Consumer Electronics, Communications and Networks (CECNet),* , Yichang, 2012, pp. 1216-1219.

[11] K. Jamsa, *Cloud Computing SaaS, PaaS, IaaS, Virtualization, Business Models, Mobile, Security, and More*. United States of America: Jones and Bartlett Learning, 2013.

[12] Abdul. (2013, March 10, 2013). *Cloud Computing And Virtualization*. Available: http://www.cloudtweaks.com/2012/12/cloud-computing-and-virtualization/

[13] L. Yunfa, L. Wanqing, and J. Congfeng, "A Survey of Virtual Machine System: Current Technology and Future Trends," in *Third International Symposium on Electronic Commerce and Security (ISECS),* , Guangzhou, 2010, pp. 332-336.

[14] A. Jasti, P. Shah, R. Nagaraj, and R. Pendse, "Security in Multi-Tenancy Cloud," in *International Carnahan Conference on Security Technology (ICCST),* , San Jose, CA, 2010, pp. 35-41.

[15] C. P.Pfleeger and S. L. Pfleeger, *Security in Computing*, Fourth ed.: Pearson Prentice Hall, 2009.

[16] S. Harris, *CISSP All-in-One Exam Guide*, Sixth ed. New York: McGraw-Hill, 2013.

[17] D. Hyde. (2009, March 17, 2013). *A Survey on the Security of Virtual Machines*. Available: http://www.cs.wustl.edu/~jain/cse571-09/ftp/vmsec/

[18] K. Owens. (2009, March 7, 2013). *Securing Virtual Compute Infrastructre in the Cloud*. Available: http://www.savvis.com/en-us/info_center/documents/hos-whitepaper-securingvirutalcomputeinfrastructureinthecloud.pdf

[19] S. Security. (2013, January 12, 2014). *VIRTUALIZATION-SPECIFIC CHALLENGES COULD THREATEN DATA SECURITY*. Available: http://www.simplysecurity.com/2013/03/05/virtualization-specific-challenges-could-threaten-data-security/

[20] K. Wood and M. Anderson, "Understanding the Complexity Surrounding Multitenancy in Cloud Computing," in *8th International Conference on e-Business Engineering (ICEBE),* , Beijing, 2011, pp. 119-124.

[21] K. Benzidane, S. Khoudali, and A. Sekkaki, "Autonomous Agent-based Inspection for inter-VM Traffic in a Cloud Environment," in *7th International Conference for Internet Technology and Secured Transactions (ICITST-2012)*, London, 2012, pp. 656-661.

[22] T. Kwok, T. Nguyen, and L. Lam, "A Software as a Service with Multi-tenancy Support for an Electronic Contract Management Application," in *International Conference on Services Computing,*, Honolulu, HI, 2008, pp. 179-186.

[23] F. Chong and G. Carraro, "Architecture strategies for catching the long tail," ed. MSDN Library, Microsoft Corporation, 2006.

[24] C. Babcock. (2011, April 8, 2013). *Cloud Security Challenges Include Audit Trails, Preventing Attacks*. Available: http://www.informationweek.com/cloud-computing/infrastructure/cloud-security-challenges-include-audit/229400272

[25] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka*, et al.*, "Controlling Data in the Cloud: Outsourcing Computation without Outsourcing Control," in *ACM Workshop on Cloud Computing Security (CCSW)*, Chicago, 2009, pp. 85-90.

[26] S. TechCenter. (2007, April 11, 2013). *Microsoft Security Bulletin MS07-049 - Important*. Available: http://technet.microsoft.com/en-us/security/bulletin/MS07-049

[27] (2009) How to Secure Cloud Computing. *Information Security Magazine*. 49. Available: http://searchsecurity.techtarget.com/How-to-Secure-Cloud-Computing

[28] H. Tianfield, "Security issues in cloud computing," in *International Conference on Systems, Man, and Cybernetics (SMC),* , Seol, 2012, pp. 1082-1089.

[29] T. Mather, S. Kumaraswamy, and S. Latif, *Cloud Secuirty and Privacy*, First ed. Sebastopol: O'Reilly Media, Inc.,, 2009.

[30] E. Shen, E. Shi, and B. Waters, "Predicate Privacy in Encryption Systems," in *Theory of Cryptography*. vol. 5444 2009, O. Reingold, Ed., ed San Francisco, CA: Springer Berlin Heidelberg, 2009, pp. 457-473.

[31] J. M. A. Calero, N. Edwards, J. Kirschnick, L. Wilcock, and M. Wray, "Toward a Multi-Tenancy Authorization System for Cloud Services," IEEE Security & Privacy Magazine, vol. 8, 2010, pp. 48-55.

[32] T. Koorevaar, "Dynamic Enforcement of Security Policies in Multi-Tenant Cloud Networks," Master of Science Department of Software and Computer Engineering, Ecole Polytechnique de Montreal, 2012.

[33] D. McKay, "Securing the virtual environment," ed: *http://www.net-security.org*, 2011.

[34] A. Azeez, S. Perera, D. Gamage, R. Linton, P. Siriwardana, D. Leelaratne*, et al.*, "Multi-tenant SOA Middleware for Cloud Computing," 2010 IEEE 3rd International Conference on Cloud Computing (CLOUD), 2010, pp. 458-465.

[35] B. Hay, K. Nance, M. Bishop, and L. McDaniel, "Are Your Papers in Order? Developing and Enforcing Multi-tenancy and Migration Policies in the Cloud," in *45th Hawaii International Conference on System Science (HICSS),* , Maui, HI, 2012, pp. 5473-5479.

[36] I. Iankoulova and M. Daneva, "Cloud computing security requirements: A systematic review," in *6th International Conference on Research Challenges in Information Science (RCIS),* , Valencia, 2012, pp. 1-7.

[37] M. Jensen, J. Schwenk, N. Gruschka, and L. L. Iacono, "On Technical Security Issues in Cloud Computing," in *CLOUD '09. IEEE International Conference on Cloud Computing,* , Bangalore, 2009, pp. 109-116.

[38] W. Hanqian, D. Yi, C. Winer, and Y. Li, "Network security for virtual machine in cloud computing," in *5th International Conference on Computer Sciences and Convergence Information Technology (ICCIT),* , Seol, 2010, pp. 18-21.

[39] D. Riresmith, "Specifying Reusable Security Requirements," Journal of Object Technology, vol. 3, 2004, pp. 61-75.

[40] R. S. Sandhu, E. J. Coyne, H. L. Feinsteink, and C. E. Youmank, "Role-Based Access Control Models," IEEE Computer, vol. 29, 1996, pp. 38-47.

[41] M. V. Thomas and K. C. Sekaran, "An Access Control Model for Cloud Computing Environments," in *2nd International Conference on Advanced Computing, Networking and Security (ADCONS),* , Mangalore, 2013, pp. 226-231.

[42] W. Wenhui, H. Jing, S. Meina, and W. Xiaohui, "The design of a trust and role based access control model in cloud computing," in *6th International Conference on Pervasive Computing and Applications (ICPCA),* , Port Elizabeth, 2011, pp. 330-334.

[43] L. Popa, M. Yu, S. Y. Ko, S. Ratnasamy, and I. Stoica, "CloudPolice: Taking Access Control out of the Network," in *ACM Workshop on Hot Topics in Networks. HotNets*, Monterey, CA, USA, 2010, pp. 1-6.

[44] X.-Y. Li, Y. Shi, Y. Guo, and W. Ma, "Multi-Tenancy Based Access Control in Cloud," in *International Conference on Computational Intelligence and Software Engineering (CiSE),* , Wuhan, 2010, pp. 1-4.

[45] S.-J. Yang, P.-C. Lai, and J. Lin, "Design Role-Based Multi-tenancy Access Control Scheme for Cloud Services," in *International Symposium on Biometrics and Security Technologies (ISBAST),* , Chengdu, 2013, pp. 273-279.

[46] C. S. Alliance. (2009, May 12, 2013). *Security Guidance for Critical Areas of Focus in Cloud Computing (2.1 ed.)*. Available: http://www.cloudsecurityalliance.org/guidance/csaguide.pdf

[47] A. W. Services. (2011, March 15, 2014). *Amazon Web Services: Overview of Security Process*. Available: http://s3.amazonaws.com/aws_blog/AWS_Security_Whitepaper_2008_09.pdf

[48] B. A. Forouzan, *TCP/IP Protocol Suite*, Second Edition ed. New York: McGraw-Hill 2003.

[49] L. Popa, "Building Extensible and Secure Networks," Phd, Electrical Engineering and Computer Sciences University of California at Berkeley, 2011.

[50] I. E. T. F. (IETF), "RFC 3697," in *Request for Comments (RFC)*, ed, 2004.