Data Confidentiality and Risk Management in Cloud Computing

Afnan Ullah Khan

Engineering Doctorate

University of York

Computer Science

April 2014

ABSTRACT

Cloud computing can enable an organisation to outsource computing resources to gain economic benefits. Cloud computing is transparent to both the programmers and the users; as a result, it introduces new challenges when compared with previous forms of distributed computing. Cloud computing enables its users to abstract away from low level configuration (configuring IP addresses and routers). It creates an illusion that this entire configuration is automated. This illusion is also true for security services, for instance automating security policies and access control in the Cloud, so that companies using the Cloud perform only very high- level (business oriented) configuration. This thesis identifies research challenges related to security, posed by the transparency of distribution, abstraction of configuration and automation of services that entails Cloud computing. It provides solutions to some of these research challenges. As mentioned, Cloud computing provides outsourcing of resources; the outsourcing does not enable a data owner to outsource the responsibility of confidentiality, integrity and access control as it remains the responsibility of the data owner. The challenge of providing confidentiality, integrity and access control of data hosted on Cloud platforms is not catered for by traditional access control models. These models were developed over the course of many decades to fulfil the requirements of organisations which assumed full control over the physical infrastructure of the resources they control access to. The assumption is that the data owner, data controller and administrator are present in the same trusted domain. This assumption does not hold for the Cloud computing paradigm. Risk management of data present on the Cloud is another challenge. There is a requirement to identify the risks an organisation would be taking while hosting data and services on the Cloud. Furthermore, the identification of risk would be the first step, the next step would be to develop the mitigation strategies. As part of the thesis, two main areas of research are targeted: distributed access control and security risk management.

LIST OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENT

First and foremost, I would like to thank God for giving me the will and the patience to complete this doctorate.

To my family:

I would like to thank my Father (Mushahid Ullah Khan), it was his idea that I should go on this path and become a doctor. I would like to thank my mother (Maliha Khan) for her prayers for me and for her love. My sisters (Rida Khan and Lamia Khan) and my brother (Farzan Ullah Khan) for their constant support throughout. My gratitude to my beloved wife (Hira Afnan) for keeping up with me for all the time that I have not been there for her. Finally, I would like to thank my daughter and my little angel Rania Khan for being a source of love and happiness for me.

To University of York, University of Oxford and BT:

This research project is carried out as part of the Engineering Doctorate in Large Scale Complex IT Systems (LSCITS) scheme at the University of York. Engineering Doctorates aim to solve research problems of clear commercial or industrial relevance and are co-supervised by both a sponsoring company and academia. In this case the industrial supervision was provided by Dr. Ben Azvine and Dr. Theo Dimitrakos of BT Plc. and academic supervision was provided by Dr Manuel Oriol of the University of York. I wish to express my thanks to all of them. I also wish to thank the Engineering and Physical Science Research Council (EPSRC) who sponsored the whole programme of LSCITS Engineering Doctorates." I also undertook 60 credits of postgraduate module at University of Oxford as part of the doctorate. I would also like to thank the admin staff and the teaching staff at the Department of Computer Science, Oxford University.

To my friends:

I would like to thank Usman Piracha, Saleem Butt and Ali Khan for their encouragement and support.

AUTHOR DECLARATION

The work in this thesis is done by the author. This work has not been submitted to any other university or this university for another award.

The following papers and patents have been published as part of the research work conducted for the EngD.

Poster Paper:

A. Khan, J. Jacob, M. Oriol, and T. Dimitrakos, "Architecture for Secure Colloborative Services," *LSCITS, Annu. Board Meet. ISAB NSB members*, 2010.

Industrial Paper:

S. K. Nair, S. Porwal, T. Dimitrakos, A. J. Ferrer, J. Tordsson, T. Sharif, C. Sheridan, M. Rajarajan, and A. U. Khan, "Towards Secure Cloud Bursting, Brokerage and Aggregation," *2010 Eighth IEEE Eur. Conf. Web Serv.*, pp. 189–196, 2010.

Workshop Paper:

M. Kiran, A. U. Khan, M. Jiang, K. Djeame, M. Oriol, and M. Corrales, "Managing Security Threats in Clouds," *Digit. Res.*, 2012.

Research Papers:

A. U. Khan, M. Oriol, M. Kiran, M. Jiang, and K. Djemame, "Security risks and their management in Cloud computing," in *4th IEEE International Conference on Cloud Computing Technology and Science (Cloudcom) Proceedings*, 2012, pp. 121–128.

A. U. Khan, M. Oriol and M. Kiran, "Threat Methodology for Securing Scalable Video in the Cloud," in *8th IEEE International Conference for Internet Technology and Secured Transactions (ICITST) Proceedings*, 2013.

Patents:

A. U. Khan, F. La Torre and M. Oriol, “Access Control and Data Confidentiality in the Cloud,” EU Patent, BT IPD Application Number: A32311, Filed March 2012

A. U. Khan, F. La Torre and M. Oriol, “Network and Access Security System,” BT IPD, EPO Patent Application Number: 13715392.0, Filed June 2013

BT’s Internal Paper:

Cloud Strategy Roadmap, Theo Dimitrakos, Srijith K. Nair, Fadi El Mousa, Afnan Ullah Khan, published internally to BT, 2011

CHAPTER 1

INTRODUCTION

1.1 CLOUD COMPUTING

Cloud computing is a new paradigm for computing infrastructure[1][2]. While Cloud computing can be thought of as just one more way of implementing distributed systems, it is different from traditional distributed systems, as well as grid computing, as its infrastructure is transparent to users and programmers alike. This allows for new ways of selling and sharing resources altogether.

Cloud computing offers a new economic model which enables enterprises to shift from the conventional way of developing their own IT departments to outsourcing their needs for software, platform and infrastructure. Cloud computing has been promoted as a new paradigm and also as the 5th utility service after water, electricity, gas and telephony [3].

A paradigm shift is taking place in the IT industry [4]. In the past enterprises used to support their business by procuring IT infrastructure and then developed their software on top of that infrastructure. Cloud computing presents a model in which IT infrastructure is leased and used according to the need of the enterprise. The fundamental benefit of this model is

that it converts capital expenditure (CAPEX) of an enterprise into operational expenditure (OPEX)[5].

We envision that this shift would enable hybrid Clouds (a combination of private and public Cloud) to become commonplace, realized by private Clouds interacting with a rich ecosystem of various different types of Cloud. We are already witnessing research being conducted to enable organisations to automatically externalise services and applications to trustworthy and auditable Cloud providers in the hybrid model[6].

1.2 RELEVANCE OF CLOUD COMPUTING TO BT

The paradigm shift that is taking place in the IT sector has opened up new avenues of growth. BT[7] being one of the largest technology company in the UK had to develop its own strategy to leverage the benefits of this shift. BT already has an extensive offering for customers requiring data centres but with the evolution of cloud computing further technological development was required.

The support for this research and development work was undertaken as part of BT's strategy to develop its Cloud offering. More specifically to develop the security offerings related to Cloud computing. BT has developed large number of patents in the different areas of IT which it uses to generate revenue. It invests a large part of this revenue to enhance growth and to add to its offerings relating to Cloud computing.

From BT's perspective the aim of the thesis is twofold. The first goal is to develop a novel scheme that would enforce access control policies on Cloud computing scenarios. The scheme will also ensure scalability to cater for large number of Cloud consumers and confidentiality of data hosted on the Cloud. We use the EU OPTIMIS[6] [8] project to perform evaluation of the scheme developed. The second goal is the development of the security risk framework for the Cloud computing platforms. This framework would provide a mechanism

through which risk analysis can be performed for Cloud computing scenarios. Details about the research plan and outcomes are given in Chapter 4.

1.3 RESEARCH CONTRIBUTIONS

The Engineering Doctorate programme envisages a 'portfolio' of research contributions within a domain that are of relevance in an industrial/commercial context i.e. that will have or may have real-world impact. The industrial sponsor for the Engineering Doctorate was British Telecom and the research presented in this thesis has been guided by current and anticipated problems faced in cloud security by BT.

Initial work reviewed both the academic state of the art in cloud computing security and the industrial state of practice. This allowed various research gaps to be identified. Background and literature review is presented in Chapters 2 (Cloud Computing Background), 3 (State of the art implementations Cloud computing implementations), and 4 (Security challenges in Cloud computing).

The specific research objectives addressed by the technical work of this thesis are as follows.

1) The production of a Risk Assessment Framework with tool support for application to Cloud computing scenarios. This is presented in Chapter 5
2) The provision of an effective and secure architecture and algorithms for secure video streaming. This is presented in Chapter 6
3) The provision of a scalable access control framework based on an existing cryptographic scheme. This is presented in Chapter 7
4) Experimental validation of the scheme developed in Chapter 7 is put forward in Chapter 8.

1.4 THESIS STRUCTURE

In this section the structure of the thesis is presented.

Chapter 2 puts forward the explanation of differences between Cloud computing and other forms of computing. The Cloud computing background and evolution chapter came from a joint paper written in BT titled as “Cloud Strategy Roadmap”.

For Chapter 3, the state of the art implementations and the comparative analysis of the security landscape of Cloud delivery models are presented. The main objective of the comparative analysis was to develop a strong understanding of the Cloud computing field. This led to the identification of security challenges for Cloud computing.

For Chapter 4, the review of the security challenges and the development of research gaps was undertaken. The identification of the research gaps setup the research direction of the thesis.

For Chapter 5, the development of the Security Risk Framework was done in collaboration with University of Leeds where Mariam Kiran was leading the risk framework as part of the OPTIMIS project. Mariam’s contribution were in the form of developing the algorithm for risk calculation. The author’s contributions came in the form of performing the threat analysis and then relating those threats with the Cloud computing scenarios of the OPTIMIS project.

For Chapter 6, Securing Scalable Video idea is presented. Part of this research work was undertaken by the researcher during his MSc at University College London. In the MSc the focus was on developing the authentication and confidentiality schemes for scalable video scenarios. In the EngD the focus remained of extending this work in Cloud computing scenarios and performing a risk assessment for secure scalable video. In this Chapter the risks were identified relating to secure scalable video in the Cloud context. The novelty in Chapter 6 is the development of a new risk model for the scalable video scenario.

For Chapter 7, the development of the scheme ACDC^3 scheme from the conception to the design and implantation is presented. The ACDC^3 scheme was verified by Francesco La Torre in form of undertaking via a mathematical formalisation and design review of the scheme. The ACDC^3 is a novel scheme which fulfils the requirements of scalability while providing confidentiality for the Cloud computing scenarios.

For Chapter 8, the experimental validation results of ACDC^3 and Security Risk Framework are presented. The development of code for ACDC^3 prototype, the test cases to verify the scheme and the comparison of the results of test cases with the standard encryption practice is presented in this Chapter.

CHAPTER 2

CLOUD COMPUTING BACKGROUND

2.1 INTRODUCTION

In this Chapter the aim is to formally define Cloud computing and then explain the subtle differences between it and previous forms of distributed computing. From an academic point of view it is important to take a step back and analyse the idea of Cloud computing critically and judge whether it is any different from other forms of computing. The subtlety that exists between Cloud computing and other forms of computing would then set the stage for further research and development in this area.

To understand the industrial perspective, the Chapter then further leads into a discussion of BT's vision for Cloud computing and how BT sees it emerging. Finally at the end of the Chapter we present the EU OPTIMIS project use cases. The research work in this thesis is based on these use cases.

2.2 CLOUD DEFINITIONS

There is no consensus on how to define Cloud computing [14] but some common characteristics exist among many of the definitions. In this section we present the definition of Cloud computing that is used throughout this report.

The most comprehensive definition of cloud computing, and the one used throughout this thesis, is given by the US National Institute of Standards and Technology (NIST):

> "*Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This Cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models*"[15], [16].

With respect to the NIST definition the characteristics, service models and deployment models of Cloud computing is discussed and explained.

The five essential characteristics of NIST's cloud model are:

- *On-demand self-service*, which means that a user can get computing resources provisioned automatically without human intervention.
- *Network access*, by which services should be provided over a network using a standard mechanism that supports multiple platforms like mobile phones and PDAs.
- *Resource pooling*, which enables pooling of Cloud provider resources among multiple tenants.
- *Rapid elasticity*, by which a consumer of Cloud services can provision resources rapidly and can scale in or scale out on demand.

- *Measured service* enables the monitoring, controlling and reporting of services to both Cloud provider and the consumer in order to ensure transparency.

There are three service models as per NIST's cloud definition, which are Software as a service (SaaS), Platform as a service (PaaS) and Infrastructure as a Service (IaaS):

- SaaS provides consumers with the capability to use applications hosted by the Cloud provider.
- PaaS provides consumers the ability to develop or deploy their own applications on the platform. The consumer however does not control the underlying Cloud infrastructure like network, servers and operating system.
- IaaS enables the consumer to provision processing, storage, network and other resources. Virtualisation is the key enabler technology for this service model which provide unprecedented flexibility to configure resources while at the same time enabling the provider to protect its underlying physical infrastructure [17].

There are four fundamental deployment models for Cloud computing as per NIST's cloud definition*:*

- *Private Cloud* is solely operated for an organization by either a third party or the organization itself.
- *Public Cloud* is available for the general public and is owned by an organization selling Cloud services.
- *Community Cloud* provides infrastructure which is shared by several organizations. *Hybrid Cloud* is a composition of two or more Clouds (community, private, public).

For this research we have chosen the NIST definition of Cloud computing as we found it the most comprehensive definition and also it is also widely used in the other research literature[18][19][20]**.**

Other definitions for Cloud computing includes Gartner's: "A style of computing where scalable and elastic IT capabilities are provided as a service to multiple customers using internet technologies" [21]. The definition from Gartner covers parts of the characteristics of Cloud computing but it does not contain references to on-demand services as well as any pay-as-you go usage model. This implies that the definition does not consider these characteristics fundamental to the Cloud computing model.

Forrester defines Cloud computing as, "A standardized IT capability (services, software, or infrastructure) delivered via Internet technologies in a pay-per-use, self-service way"[22]. This definition does not cover the platform part of Cloud computing paradigm. This implies that it does not make a distinction between the PaaS and the IaaS. Furthermore, It is also not clear what they mean by "self service way".

The 451 Group defines Cloud computing as, "a service model that combines a general organizing principle for IT delivery, infrastructure components, an architectural approach and an economic model – basically, a confluence of grid computing, virtualization, utility computing, hosting and software as a service (SaaS)"[23]. An important distinction this definition makes is that of confluence of Grid computing with virtualization and other Cloud computing related technologies. This distinction is very insightful as it sheds light on the influence of Grid computing over Cloud computing.

The definitions mentioned above cover many technologies and various models. We can clearly see that there is no consensus among them [14].

2.3 UNIQUE CHARACTERISTICS OF CLOUD COMPUTING

We have provided numerous definitions of Cloud computing in section 2.2, now in this section an effort is made to identify the subtle differences of Cloud computing with Grid computing.

Cloud computing came forward as the natural evolution of Grid computing, although it has been questioned whether it offers anything new or it is just rebranding of the old ideas[14][1]. The vision of Cloud computing is the same as that of Grid computing, that is to reduce cost of computing, increase reliability and increase the flexibility by transforming computers into something that we buy from a third party and then pay per use[14].

One of the major differences between the business model of Cloud computing and Grid computing is that the Cloud is transparent to the users and programmers whereas Grid follows a project oriented approach. In Cloud computing anyone can go online and get access to immense computing resources and pay only for the resources that they consume. In Grid computing the focus is project oriented. Users or a community subscribe to a Grid project to offer resources or consume resources depending upon their need. There is an effort to develop a Grid economy using Grids all over the world. The Grid infrastructure would offer services such as trading, negotiation and allocation of resources[24].

For resource management most Grids use a batch model where users submit batch jobs. These batch jobs are placed in queue. The jobs are executed with respect to the requirement specified by the user for example the job would run on 100 processors for 1 hour. The Cloud computing batch model is different as the jobs are run concurrently and the resources are shared by the users. This introduces security issues relating to data confidentiality and access control such as data leakage to unauthorised users.

Cloud computing uses virtualization to achieve abstraction and encapsulation. Virtualisation creates the illusion that many users jobs are running at the same time by creating a fabric of compute, storage and network resources. In Grids each participating organization maintains full control over its resources (i.e by not virtualising them), therefore there is less reliance on virtualization compared to Cloud computing [14]. As in Grid computing an organization maintains full control over their resources, the security requirements that are developed to cater for the needs of the Grid computing differ from those of Cloud computing. In Cloud

computing the control is lost by the data owner, as no physical control is available when it hosts data on the Cloud server. This change induces security issues relating to regulatory compliance, loss of privileged access, data confidentiality, access control etc.

Cloud computing differs from Grid computing in many ways. The business model of the two forms of computing is completely different. Cloud computing is offered on a pay per use model whereas Grid is offered in the form of a project. Moreover, the delivery models of Cloud computing (SaaS, PaaS, IaaS) differ from Grid computing delivery model which is focused towards computing power. These differences have their impact on security of Cloud computing and we discuss in more detail the threats relating to Cloud computing in Chapter 3.

Having explored the nature of Cloud computing and how it differs from Grid computing, we now explore Cloud computing from a technical perspective.

2.4 CLOUD ECOSYSTEM

There is no long-term qualitative market differentiation for Cloud providers. For example, Amazon EC2 [25] was the first to propose solutions for the Infrastructure as a Service (IaaS) but it was soon followed by major competitors like Google and IBM offering IaaS platforms like Google Compute[26] and IBM IaaS[27] at almost the same price. In order for a vendor to make a difference, it is necessary to constantly innovate and develop its offering.

Most of the Cloud services until now are provided by infrastructure providers (Cloud islands) such as Amazon EC2[25], Google App Engine[28] etc. Recently, new technologies such as Vsphere[29] lead a transition from incompatible solutions provided by Cloud vendors (Cloud Islands) to solutions that can run on several Clouds (Virtual Clouds). The assumption here is that different hypervisors (Vsphere, HyperV, Xen) will provide functionality to interconnect. The idea is that the Virtual Cloud consumes the services of the Cloud islands by developing

a virtualization layer on top of them and frees their customers from being locked into a particular IaaS vendor.

Cloud computing is also witnessing a transition towards open source platforms such as Openstack[30]. Openstack offers open source software for creating private and public Clouds. HP has consumed Openstack to build its Cloud platform called HP Cloud[31]. More vendors are now focusing on Openstack offerings (Paypal, Wellsfargo) therefore starting new era where Cloud islands can now communicate horizontally using open source software. Vsphere is a product of VMWare. Although it provided API level access to a hypervisor it failed to gain traction from other companies, possibly for reasons of competition and lack of trust. Openstack seem well on course to achieve the transition from Cloud Islands to Cloud Horizontal Federations.

In figure 1, we show what we can anticipate for the evolution of Cloud computing. The first phase was the development of 'Data Centres' [82] for the purpose of storage and computing. In the second phase we have seen the development of 'Virtual Data Centre' [7] that provides full functionality of a Virtual LAN for an organisation. A VLAN can be defined as a logical network that maps workstations and servers on some other basis than geography. It is a logical segmentation of network rather than physical one. Now we anticipate that the next step will be the emergence of 'High-end Cloud Environment' where many Virtual Data Centres will be federated to provide services. This will enable the creation of a market place for Cloud platforms and the brokers will be able to resell services to clients depending upon their specific requirements.

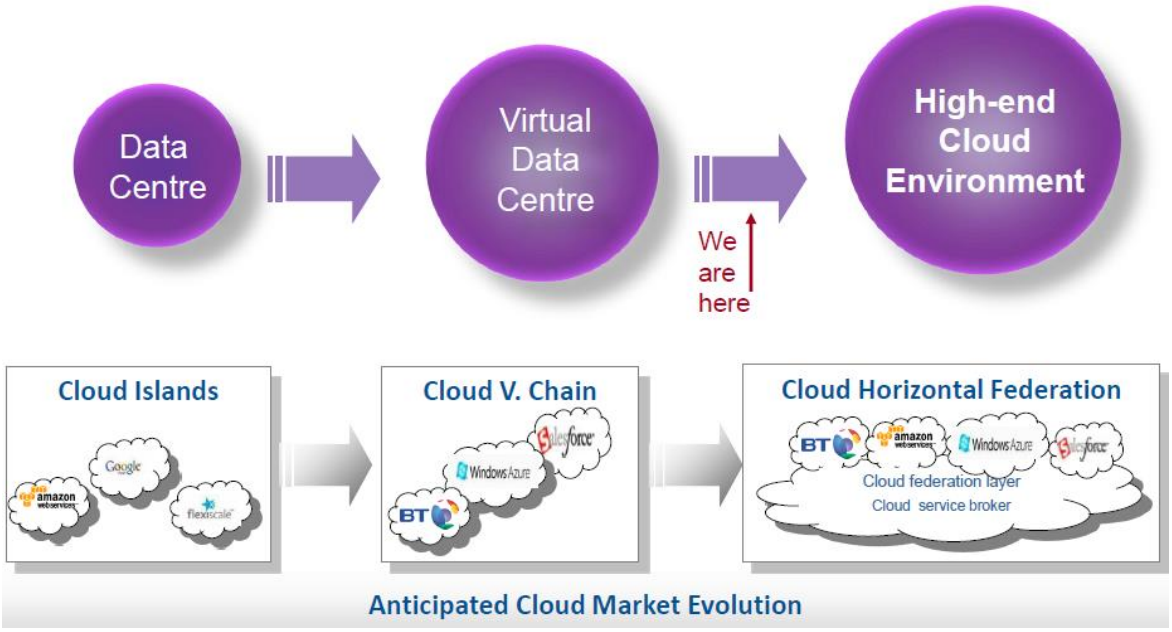

Figure 1: Anticipated Cloud Market Evolution

Currently we are in the evolution process where virtual data centres are being offered by various vendors such as BT [7].

The emergence of Cloud-aware application design patterns is making the development of Cloud based applications more convenient. Rather than focusing on programming, the concept is to now to focus on the idea. Furthermore the improvement of the instrumentation offered by the standardised interfaces of both Cloud infrastructure and Cloud platforms make application development and deployment more convenient.

In the next sections we expand the Cloud scenario that we will be using for this thesis. We detail two use cases to further explain the cloud computing scenario. The first use case is that of an "Enterprise using multi Clouds"[8] and the second use case is of "Enterprise Cloud broker"[32].

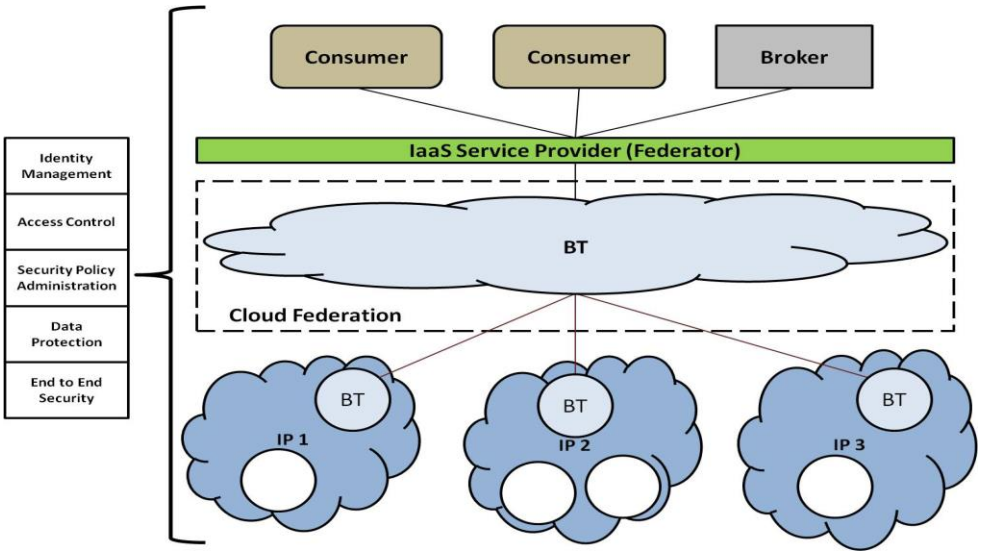

Figure 2: Virtual Cloud Scenario

2.5 CLOUD SCENARIO

The purpose of this section is to describe the Cloud computing scenarios and use cases that will be followed throughout this thesis. The explanation of use cases and scenarios now (instead of, for example, before the assessment of the commercial state of the art) will give some context to the research work, and to indicate some typical uses for cloud computing technology.

In Figure 2, a Cloud scenario is put forward where the Virtual Cloud federates the infrastructure providers (IP1, IP2, IP3) by deploying a virtualization layer on top of the infrastructure providers. This virtualisation would be able to offer IaaS services to the consumer.

The actors of the scenario are described below.

Service Provider (Federator): The federator is responsible for creating the abstraction layer that would combine the resources provided by different infrastructure providers. The design

of the federator is such that the infrastructure provider should not be aware that its services are resold to the customers. To achieve this, the federator creates virtual LAN or an overlay network that would connect the resources of these infrastructure providers. A virtual LAN creates a logical network which is not based on geography. Rather in this case it would be based on compute or storage services put together for a consumer. This creates the horizontal federation of infrastructure which would enable the federator to introduce interdependencies between the virtual machines that exist on different infrastructure providers.

Consumer: The consumer can either be the enterprise customer who can be a company or an individual user trying to access the federator in order to utilize Cloud services.

Broker: The Broker sits between the infrastructure provider and the service provider or between the federator and the customer. The Broker offers value-added services between the infrastructure provider and the federator, such as performance prediction (based on monitoring of historical data), security and SLA negotiation. A broker creates a governed and secure Cloud management platform to simplify the delivery of complex Cloud services[32]. The broker is explained further in the use case 2 explanation in the section 2.5.2.

Infrastructure provider: The infrastructure provider provides services to the federator like storage, computing and other computing resources.

We further extend the scenario by explaining two use cases. These use cases will then be used to perform the threat analysis on the scenario. The threat analysis will therefore lead us to identify security related issues and challenges.

2.5.1 USE CASE 1: AN ENTERPRISE USING MULTIPLE CLOUD SERVICE PROVIDERS

In this use case, an enterprise combines infrastructure services provided by multiple infrastructure providers to implement and realize an internal process. For example, the

process could be a one-off simulation of a complex data set that uses multiple infrastructure providers, e.g., Amazon EC2 for the compute-intensive tasks, NVIDIA Reality Server[33] Cloud service for graphics-intensive tasks and Amazon S3[34] for data storage intensive tasks and then the enterprise has to perform the necessary merging on the result set to generate the desired results. Amazon does provide services where a complete Cloud can be created using multiple tools through which Amazon EC2 and other Amazon services can be combined. However, there is absence of an interoperability layer between different vendors. Therefore the above description can only become a reality if NVIDIA provides an API level access to Amazon services. Even if that API level is present, it would still be required to orchestrate the whole process. This is where the service integration layer comes in as explained in the Figure 3.

In figure 3, we use a virtualized services integration layer that uses well-defined interfaces and standardized calls between different web-service enabled applications and data sources to provide the loosely-coupled integration required for the completion of the enterprise's process.

The service integration layer would require an interface for services such as authentication of the consumers, access control to ensure access is granted appropriately and finally federation of multiple Clouds. This federation would require the communication interface through which different services from different vendors can be orchestrated.

The multi Cloud use case is designed to realise the benefit of using different Cloud platforms at the same time as an integrated service. The challenge that this use case addresses is that of integrating different platforms.

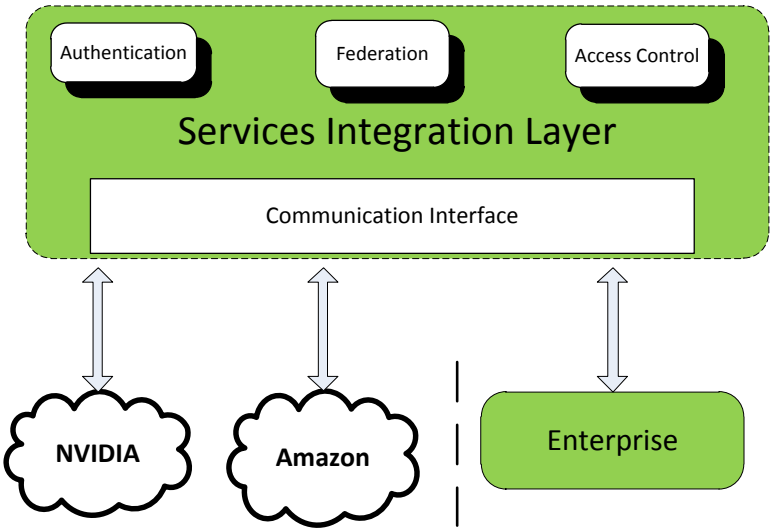

Figure 3: Enterprise using multiple Infrastructure providers

2.5.2 USE CASE 2: AN ENTERPRISE CLOUD BROKER

In the previous use case the focus was on integrating the multiple different platforms. In this use case the focus is on developing a broker service that can negotiate price, SLA etc. on behalf of different customers.

In figure 4, an enterprise approaches a Cloud broker with a given set of functional requirements and constraints. Depending upon these functional requirements and constraints the broker then picks up the best available match of infrastructure providers. These requirements and constraints can be cost, SLA parameters and other non-functional requirements like audit, compliance and security capabilities. In addition to helping in choosing the most suitable Cloud service for the enterprise's needs, the Cloud broker should also help in integration of the enterprise's processes and their deployment on the target platforms of the infrastructure provider.

Now the Cloud broker could in principle be doing a comparative analysis of services provided as in the case of use case 1. The technical challenge in use case 2 is that of performing comparative analysis between different IaaS providers or doing comparative analysis between different service providers as in the case of use case 1.

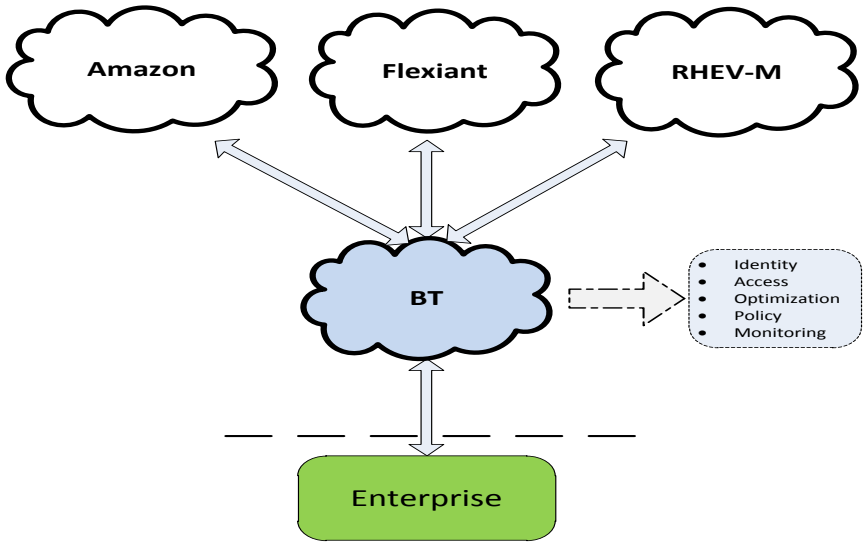

Figure 4: Cloud Brokerage

The process of sorting through the services provided by different IaaS providers is a complex process and the Cloud broker provides additional value-added and middleware-oriented services to achieve its functional requirements.

In the next Chapter we analyse the state of the art implementations of Cloud computing. A comparative analysis of the state of the art is also presented. The purpose is to undertake a thorough review of the implementations in order to develop a comprehensive understanding of the state of the art from an industrial perspective. This understanding would later enable the development of further research in the area of Cloud computing security.

CHAPTER 3

STATE OF THE ART CLOUD COMPUTING IMPLEMENTATIONS

3.1 CURRENT CLOUD SOLUTIONS AND THEIR SECURITY MODELS

This section presents current Cloud solutions and their security model. We categorise the solutions into: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). These categories are based on the Cloud service models presented in Chapter 2 which are based on the NIST definition of Cloud computing[35]**.** Furthermore, we also analyse the security of vendors which provide Cloud based security services, these services generally described as Security as a Service (XaaS).

The reason we use this classification is because the focus of the security controls would be different for each service model. For instance SaaS would focus more on the web-security

controls whereas the IaaS would be more tuned towards protecting hypervisor and VM isolation. However, numerous security controls would overlap, as every provider would have to deploy security controls to protect customer data.

For reviewing the state of the art implementations of Cloud computing, we use publically available data such research articles, security data sheets and white papers provided by companies that provide Cloud services. The survey may not be able to capture a full understanding of the architecture of the products, as the companies do not publish all data. The companies that we have selected for our survey would cover the full spectrum of Cloud delivery models such as IaaS, PaaS and SaaS. Moreover, we have also reviewed the security standards (ISO, PCI etc.) that these companies conform to in order to understand the maturity of security controls that they have employed.

3.1.1 IAAS

There are currently five major companies who provide IaaS: Amazon[25][34][36]**,** Rackspace [37][38], Joyent [97], GoGrid [39] and Terremark[40]. The security features that we have considered for our analysis are provided by these companies and are documented via publically available information. The companies which have been selected for this survey are broadly considered market leaders in the cloud computing domain. Moreover, those we are analysing for this survey tackle prominent challenges in Cloud computing relating to security. We also use some benchmark standards of IT Security to develop our analysis, these are Payment card industry data security standard (PCI-DSS), ISO standards, SAS standards etc.

Security Features /IaaS providers	*Amazon*	*Rackspace*	*GoGrid*	*Joyent*	*Terremark*
Security Model	Shared responsibility model	Shared responsibility model	Not mentioned	Shared responsibility model	Not mentioned
Confidentiality of Data	Allow customers to encrypt their own data using their own keys	Encrypting data before it leaves customer premises using up to 256 bit AES keys[41]	Not mentioned	Allow customers to encrypt [42]	Not mentioned
Hypervisor level security	Using customised version of Xen [43] to perform separation of hypervisor with the guest OS	Not mentioned	Not mentioned	Yes	Not mentioned
Access Control	Not mentioned	Access control mechanism	RBAC[41]	Yes	Not mentioned

		implemented			
Network level security	SSL/TLS	VPN[37]	VPN	Yes	Not mentioned
Accreditations / Compliance	SAS70 type 2 audit [44]	SAS type 2 audit, ISO 17799, PCI-DSS	SAS type 2 audit , HIPPA, PCI- DSS	Not mentioned	PCI DSS
Multi factor authentication	Yes	Not mentioned	Not mentioned	Not mentioned	Not mentioned
Data leakage	Amazon claims that customer instances have no access to raw disk. Data Leakage Prevention (DLP) is not mentioned in the documentation.	Not mentioned	Not mentioned	Not mentioned	Not mentioned

Intrusion detection systems	Yes	Yes	Yes	Not mentioned	Yes

Table 1: IaaS Comparison

Table 1 [25][34][36][37][38][97][39][40], shows that most of the infrastructure providers use the same security model: 'shared responsibility'. In the shared responsibility model the security responsibility is shared by both the infrastructure provider and the user of the service. The reason is that infrastructure providers have control over the layers on and below the hypervisor. They can only provide limited level of control when it comes to the VM itself. Therefore the infrastructure providers take responsibility for the hypervisor and the layers below whereas the responsibility of the VM remains with the customer. The customer has generally have admin level control over the VMs therefore enabling him to access core services of the operating system. In some cases the IaaS allows the customer to upload its own template of operating system.

Another common feature among the vendors is that they provide controls to ensure the confidentiality of data when stored or on the move. In order to ensure compliance, there are a number of accreditations that the infrastructure providers comply with. These accreditation provide limited security controls and none of them are tuned for the Cloud computing scenario. Like PCI DSS is a standard used by the payment industry, it defines the controls that should be in place in order to make the payment systems secure. Another feature that is almost common among the infrastructure providers is the inclusion of an intrusion detection system.

Joyent provides special tools that can be used to provide an extra layer of security at the VM. Amazon is the only vendor that offers key rotation service. Key rotation enables a customer to change encryption keys. Customers provide a certain number of keys to perform

encryption; these keys are then used by Amazon to perform encryption on their data. The keys are revolved with respect to the time setting, for instance the first key would be revoked after one week.

For the IaaS, we have analysed most of the major vendors in the market. Amazon provides by far the most detailed documentation with respect to security controls whereas Terremark only identify a few security controls for their IaaS offering. Almost all of the vendors provide controls relating to network level security and intrusion detection systems. On the contrary, no vendor mentions controls relating to data leakage prevention.

3.1.2 PAAS

In this section we review the solutions relating to Platform as a Service (PaaS) from the security perspective. The companies that provide PaaS are Windows Azure[45], Force.com[46][47]**,** Google Apps Engine [28] and Heroku [48]**.** The security features that we have picked for the analysis are the ones mentioned in the documentation of the companies.

PaaS providers /Security features	Windows Azure	Force.com	Google Apps Engine	Heroku
Security Model	Responsibility on the vendor	Responsibility on the vendor	Responsibility on the vendor	Not mentioned
Confidentiality of Data	Data encryption is provided as an optional feature. Internally the	Data encryption using 128 bit keys for symmetric and 1024 bits for	Data is not stored in clear text	Not mentioned

	Azure platform encrypts data.	asymmetric encryption.		
Compliance	Safe harbour	Safe Harbor agreement, Sarbanes-Oxley, HIPPA	Safe Harbor agreement, Sarbanes-Oxley, HIPPA	Uses Amazon AWS [49] at IaaS level so similar compliance
VM security	Least privilege policy is in place. Customers not provided admin access to VM. VM and VLAN isolation is done using packet filtering	Not mentioned	Not mentioned	Not mentioned
Network level security	SSL/TLS	SSL/TLS	SSL/TLS	Not mentioned
Accreditations	ISO 27001	SAS70 type 2 audit, ISO/IEC 27001	SAS70 type 2 audit, NIST SP-800-61	Uses Amazon AWS at IaaS level so similar compliance

Redundancy	Multiple backups are created	Multiple backups using RAID disks and redundant configuration of servers. Third party services are acquired for disaster recovery	Multiple backups on geographically distributed locations	Not mentioned
Data deletion	References are deleted	References are deleted, No formatting is done	References are deleted and other customer data is overridden	Not mentioned
Access control	Two models for access control are provided	Access control mechanisms are in place	Not mentioned	Not mentioned

Table 2: PaaS Comparison

Table 2 shows that most of the vendors provide features for addressing security requirements. For confidentiality most of the vendors are using the Advanced Encryption Standard [50]. The key length range is 128-256 bits. SSL/TLS is the commonly used tool for the confidentiality of network level traffic.

For compliance most of the vendors are also complying with bills of the parliament, acts ad rules like Sarbanes-Oxley[51], Data Protection Act[52] and HIPPA[53], [54]. Apart from

compliance there are other accreditations that the industry uses to ensure adequate security controls are in place such as ISO 27001 and SAS type 2 audit.

For redundancy multiple backups are created by all the vendors and third party controls are put in place for disaster recovery. For *data deletion*, all the vendors only delete the references and do not perform any formatting on the data. The data is then normally overridden by other customer data. The references to the data are deleted but not the data itself, there is a risk that customers can potentially recover deleted data from other users using advance data recovery tools.

Access control is also provided by most of the vendors but the level of control varies.

Windows Azure is the only platform that explicitly mentions in the documentation that they deploy security controls to ensure the VLANS and VM remains isolated. Controls which they use are packet filtering and access control. The isolation of VLANS and VM ensures that a customer having control over his VM cannot penetrate into the VM or VLAN of another customer using the same Cloud infrastructure.

Windows Azure also only provides confidentiality to the customer as an additional feature although all internal traffic is kept confidential. Windows Azure security provide adequate security controls to tackle the threats of identity, access management, confidentiality and network level security[55]**.** Security features relating to data leakage prevention and usage control threats are not mentioned.

All data hosted by the Google Apps Engine is stored in encrypted format. Therefore confidentiality of data is ensured by Google. The issue of access control is tackled by applying access control policies such as access is given on a need to know basis. Logs are maintained and all activities are monitored to ensure access control policies are adhered by the Google staff. The access to the production environment is further controlled by a centralised access control mechanism.

Force.com provides security features similar to the ones provided by its SaaS offering which is Salesforce [56].

Heroku uses Amazon AWS [48] as the underlying IaaS offering. This implies that it has the similar level of security controls that Amazon AWS provides. We discussed in section 3.1.1 the Amazon EC2 related security controls which also uses AWS.

For PaaS, we have reviewed all the major vendors in the market. Most of the vendors provide similar levels of security controls for data confidentiality, redundancy and network level confidentiality. Microsoft provides an extra level of security controls which are VM isolation and access control. On the contrary, Heroku do not specify security controls for its PaaS offering and it relies completely on the IaaS provider for security.

3.1.3 SAAS

We review all the solutions relating to Software as a Service (SaaS) from the security perspective in this section. The major companies that provide SaaS are Salesforce [57], Rackspace email & Apps[37], Marketo [58][59] and Zuora [60]**.** Marketo specialises in marketing automation software. Zuora provides a subscription business model to its customers and its services ranges from finance, commerce and billing. The companies selected for SaaS delivery model are leaders (ranked by Inc. 500 and Fortune 100) in their specific SaaS domain, this gives the analysis a thorough mix of what is available in the market and what kind of security services are provided by the SaaS providers. The security features that we mention in the analysis come from the documentation of the vendors that we are using.

SaaS providers/ Security features	Salesforce	Rackspace email & Apps	Marketo	Zuora

Security model	Responsibility on the vendor	Responsibility on the vendor	Responsibility on the vendor	Responsibility on the vendor
Single Sign on	Supported	Supported	Not mentioned	Supported
Network level confidentiality	SSL/TLS	VPN	SSL/TLS [129]	SSL/TLS
Confidentiality of data	Data encryption using 128 bit keys for symmetric and 1024 bits for asymmetric encryption	Encrypting data before it leaves customer premises using upto 256 bit AES keys [61]	AES encryption, customer data stored in separate databases	Not mentioned
Multi factor authentication	Not mentioned	Not mentioned	Yes	Not mentioned
Accreditations	SAS70 type 2 audit, ISO/IEC 27001	SAS type 2 audit, ISO 17799, PCI-DSS	SAS type 2 audit	Not mentioned
Compliance	Safe Harbor agreement, Sarbanes-Oxley, HIPPA	Not mentioned	Safe Harbor agreement	Not mentioned

Table 3: SaaS Comparison

Table 3 shows that most of the vendors take care of the security responsibility. At the network level most of the vendors use SSL/TLS whereas for encryption of data the most common algorithm is that of AES.

Single Sign On (SSO) is supported by most of the vendors whereas multifactor authentication (multiple criteria's are used for authentication) is only supported by Salesforce. In [62], Armando using formal models has revealed a severe security flaw in the SAML (industry standard for SSO) based protocol used by Google Apps Engine that allows a dishonest service provider to impersonate a user at another service provider.

For accreditations and compliance the industry standards are used by most of the companies, standards include Safe harbour agreement and SAS type 2 audit. Safe Harbour agreement enables the movement of personal data between the EU and the US. The companies which want to move personal data from the US into the EU region or vice versa need to comply with criteria set out in this standard to gain eligibility for moving personal data of US and EU citizens.

Salesforce and Rackspace email and apps use the IaaS offerings from Force.com and Rackspace respectively who are the IaaS offerings of their sister company. For the SaaS they further hardened the security by providing layer of authentication and SSO. The rest of the security controls are the same as provided by Rackspace which we have mentioned in 4.2.1

For SaaS, most of the security features (SSO, Multi-factor authentication, SSL,VPN) provided are from the perspective of web-security as most of the services are provided over the web.

3.1.4 SECURITY AS A SERVICE (XAAS)

In this section we analyse the security of services provided by Cloud based service providers. For this analysis we take into account companies which are less known and are smaller in

size. Moreover, we also look at the level of security provided by the market leaders in this area.

The market for Cloud based security services include Remote vulnerability assessment, Identity & access management (IAM), Web gateways, Email gateways etc. This market is growing at a very fast pace and is expected to go from being $2.13 billion in 2013 market to $4.13 billion by 2017[63].

For the analysis we have selected Qualys[64] which is a market leader in providing Cloud based vulnerability assessment and penetration testing. CipherCloud[65] is another company that specialises in providing security services for Cloud and has distinguished itself by providing services such as searchable encryption, Cloud data loss prevention, encrypting emails etc. Radar Services is company that specialises in providing Cloud based Security incident and management solution (SIEM). The company is small in size but has been growing steadily. The reason we selected Radar Services was because Cloud based SIEM is an emerging domain for XaaS and there are not many large enterprises providing this service.

XaaS providers/ Security features	Qualys	Radar Services	CipherCloud
Security model	Shared	Shared	Shared
Single Sign on	Supported (SAML 2.0)	Not mentioned	Yes with Salesforce
Network level confidentiality	SSL/TLS	SSL/TLS	SSL/TLS

Encryption	N/A	Not mentioned	AES 256 bit encryption[66]
Key Management	Supported (Keys to be provided by the consumer)	Not mentioned	Yes
Malware Detection	Yes	Yes	Yes
Data Leakage Prevention	Yes[67]	Not mentioned	Yes

Table 4: XaaS Comparison

For all the companies providing XaaS, it is expected that the security model is shared between the XaaS and consumer. The primary role of the XaaS is provide a security service which is specific to a domain like DLP. Therefore, the consumer would be responsible for ensuring that it provides appropriate access to XaaS to its systems. Moreover, if there are breaches at levels not scanned by the XaaS then the responsibility will be with the consumer.

3.1.5 SECURITY FEATURES AND VENDORS

For all three service models (IaaS, PaaS, SaaS), the documentation of companies do not provide detail information on the security features that are provided. Normally large companies like Microsoft and Google have detailed documentation available but even they do not provide the full spectrum of the security features applied. This is because it can reveal security vulnerabilities of their system which may in turn be exploited by the attackers.

As for the other smaller companies like Heroku, Zuora, Radar Services etc. they do not mention even the basic security features like data confidentiality, access control, hypervisor security and VM security.

The focus of Cloud providers is shifting towards providing more advanced services for securing Cloud infrastructure. These services include Security incident & response (SIEM), Identity and access management (IAM), Cloud based SSO, Encryption as a service etc. This trend of providing security through a Cloud service is usually termed Security as a Service (XaaS). We have made an effort to cover this domain and comparative analysis of three companies are provided.

The next Chapter focuses on research challenges that arise within the domain of Cloud computing security from an academic perspective. A thorough literature review is conducted which leads to identification of research gaps.

CHAPTER 4

SECURITY CHALLENGES IN CLOUD COMPUTING

The purpose of this Chapter is to identify the research gaps within the area of Cloud computing security by analysing the state of the art and by categorising the research area. The approach that is taken to identify the research gaps is by conducting systematic analysis of the literature. Furthermore, the Chapter also puts forward a research agenda that is to be followed to fill the research gaps.

4.1 ACCESS CONTROL IN CLOUD COMPUTING

Cloud computing has been widely adopted by the industry (Telecoms, Banks, Government etc.) and according to IDC its market was worth $16 billion in 2008 and by 2018 the market will rise to $127.5 billion (SaaS ($82.7 billion), PaaS ($20.3 billion), IaaS ($24.6 billion))[68].

From the technological point of view even with such a large market size, Cloud computing is still in its infancy[69].

Security is however a priority concern for many Cloud computing customers who make buying choices on the basis of the reputation for confidentiality, integrity and resilience, and the security services offered by a provider. This then is a strong driver for Cloud providers to improve their security practices and compete on security services[70]**.** In the previous chapter we showed that many vendors do not publicise their information security measures. The reasons for this are not clear. They might wish to keep these measures secret but it is also possible that they do not have a mature security model implemented within an organisation. However, for Cloud consumers, security remains a top concern.

Data security for Cloud computing has been identified as one of the major research challenge by Zhang[71], Kandukuri [72]**,** Shen[73]**,** Popović[74] and Catteddu [70]**.** One reason why Cloud computing is considered such as major adoption challenge is because data security can be a show stopper.

Customers do not have physical access to data that is stored on the infrastructure provider premises. They have to trust the infrastructure provider with respect to the confidentiality of the data. The infrastructure provider has to build in security controls that would ensure that data remains confidential when it is stored and when it is on the move.

Cloud computing follows a multi-tenant architecture where data from different customers are stored on the same server. Though this model is very economical, it entails many security risks. Segregating data stored from different customers is very important: if not properly implemented, this can lead to one customer accessing the data of another customer.

Most of the commercial offerings only delete the references associated with the data rather than deleting the data itself. This means that if data recovery tools are used, one customer can potentially recover the data of another customer. This sort of problem has been known

for a long time. Indeed the earliest major security standard of the 1990's [75] identifies "object reuse" as a specific area of security functionality.

Infrastructure providers should build in security controls that would ensure that even though the infrastructure is shared at the physical level and the software level, the data is segregated. Security can be a show stopper for infrastructure providers, if a data breach occurs this would lead to loss of confidence, therefore leading to Cloud consumers moving services out of that particular Cloud provider infrastructure. Sharing of hardware is a necessity that is required for Cloud computing however, segregation of data is a software requirement that would enhance Cloud consumer confidence in the infrastructure provider.

At the IaaS level the segregation of data is even more important as customers have control over the virtual machine and they can install and run software. This gives them the freedom to attack the system; we have witnessed an attack on the EC2 platform in[76].

Another attack on Amazon S3 happened on June 2008 when customers using the storage facility said that data has been corrupted and the checksum of the data was failing[77]. Furthermore Amazon clearly states that it delegates the risk associated with data to the data owner who should take precautionary measures.

From the customer's point of view it is important that Cloud providers have mechanisms in place that gives them the confidence that the data stored on the Cloud is not tampered with. Failing to do so means that customers cannot trust that the data is securely hosted.

One of the most important security controls that is used to ensure data security is access control. Access control in an enterprise is performed by using specialised software that grant access on the basis of roles or attributes and the security policy they hold corresponding to these roles. The data on Cloud platforms are stored in many different locations of which the customer may not be even aware.

The main challenge is to prove legally or technically that unauthorised access has taken place in the Cloud. As the infrastructure provider controls the Cloud, we need to investigate how a Cloud consumer can use technology to ensure that unauthorised access does not take place at the infrastructure provider level. This mechanism if provided to Cloud consumers can lead to further augmentation of confidence in the infrastructure provider. Moreover, from a legal perspective it would useful to register if a breach has taken place or not. This is necessary to comply with standards such as that of HIPPA, Data Protection Act etc.

4.1.1 RESEARCH CATEGORISATION

For clarity we have categorised the research papers that we have analysed as the state of the art with respect to access control, scalability and user revocation. This categorisation has been made after reviewing clusters of research activity while analysing the literature. The degrees of scalability and access control provided give two axes of categorisation. The following chart shows the categorisation,

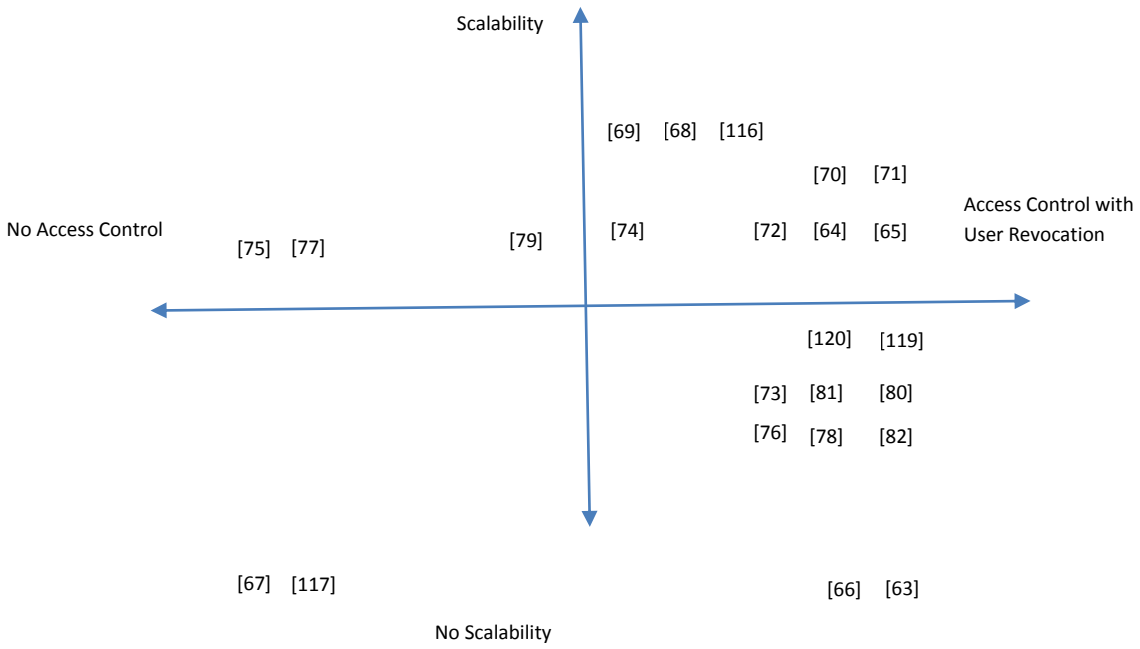

Figure 5: Research Categorisation

Figure 5, puts forward the requirement for a scheme that solves the problem of scalability and access control with user revocation for Cloud computing scenarios.

4.1.2 TRADITIONAL ACCESS CONTROL

In Discretionary access control (DAC) an individual subject can control the decision of an access control mechanism so that it allows or denies access to an object[78][79]. The approach is also known as the identity based access control, as the identity has control over the access control decision. DAC is widely supported by operating systems such as UNIX and Microsoft Windows. In Mandatory access control (MAC) the system mechanism (normally the operating system) controls access to an object and individual identity or subject cannot alter that access [79][80]. MAC depends on the correct classification of subjects and objects. MAC uses the Bell-LaPadula confidentiality model which has two security properties. Firstly, the simple security property ensures that no process may access resource labelled with higher classification. Secondly, the military classification property prevents processes from

writing to a lower classification. These two properties makes the MAC inflexible. In a system where there are large number of subscribers who are accessing numerous resources, to maintain the MAC properties require significant computing resources therefore making MAC difficult to scale.

Both MAC and DAC are heavily used in operating systems. Using these models for applying access control for VMs via the hypervisor has already been achieved. In fact an example of application of MAC can be witnessed in the hypervisor sHype[81][82] [14] [15].

Access control lists[83] are a simplified access control mechanism by which a table is created where an access rights list is maintained. Normally there are two columns in the table one containing the subject and the other containing the access rights. An ACL is linked with resources such as files or folders and decision with regards to access control is performed using these lists. ACL are usually used in centralised environments where users control their own file security. ACL's are less suited in environments where the user population is large and constantly changing. Furthermore, ACLs are not an optimal choice when it comes to performing security check at runtime. Typical operating systems keep track of which user is running which program, rather than which files has been authorised to access[84]. The problems of scalability and trusted domain (owner and controller of data to be in the same trust domain) are shortfalls of ACL when it comes to Cloud computing.

The basic concept of Role Based Access Control (RBAC)[85] [16] [86] is that users are assigned roles and permissions are associated with the roles. The users acquire access by becoming members of the roles. The relationship that exists between roles and permissions can be many to many. Similarly the relationship between users and roles can be many to many. Therefore the same user can have many roles and a role can be assigned to many users. RBAC requires the identification of roles within a system, as a role represents the authority and responsibility of the system users.

The traditional access control models implementation has focused on the assumption that data controller and data owner is in the same trust domain, the assumption does not hold for Cloud computing. The Data controller in this scenario is an organisation having administrative and physical control over the infrastructure hosting the data.

In the Cloud computing environment, the main challenges faced by RBAC scheme are scalability, authentication, delegation of authority and trust. There are some RBAC schemes that have been developed for Cloud computing environments. Younis et al. [87] put forward a scheme which caters for some of the requirements relating to the Cloud computing environment. It does not, however, convincing tackle the problem of scalability. The scheme AC3 that they present in the paper is based around RBAC. RBAC relies on roles, as the number of Cloud users increases, the number of roles augments exponentially. The inherent problem of scalability for RBAC is still present in the scheme which is that it becomes extremely difficult to manage for large number of users.

Singh et al.[88] presents an upgraded form of RBAC for Cloud computing scenario. The scheme focuses on tackling issues relating unauthorised access by putting restriction on number of roles assigned per user. Moreover, it keeps a backup of all authorisation data on local stores to have redundancy.

These approaches still do not fundamentally addresses the issue of scalability of RBAC within Cloud computing scenario. The issue remains that in the number of Cloud consumers rises, the number of roles rises significantly. Managing the roles and permissions becomes increasingly difficult and expensive. Moreover, this scheme relies on making backups of authorisation data which further makes it difficult to scale.

4.1.3 DISTRIBUTED ACCESS CONTROL

Breach of an access control policy may lead to the loss of confidentiality of data. As shown in the threat analysis in Chapter 3, loss of confidentiality of data and enforcement of access

control is rated as one of the most high level risks(most serious) associated with Cloud computing.

A general solution to providing access control in a distributed scenario is to encrypt the data that is outsourced to a server or Cloud platform. The users who have the right credentials will then be able to access that data. This approach has been adopted by many systems [89] [90] [91] [92]. The problem with this approach is that it adds to the complexity because of management of keys and encryption of data. As Cloud computing is both transparent to users and programmers, it would be used in a highly decentralised environment with huge numbers of users. The goal is to achieve scalability and fine grained access control without introducing a very high level of complexity. For instance, if a system is dependent on PKI then it would have issues of scalability, as PKI does not scale due to technical and social issues. Furthermore, keeping the security policies concurrent (and ensuring consistency) when data is residing on many distributed platforms is also required.

In the rest of this section, the description of the state of the art and shortcomings are explained.

Kallahalla et al. [89] propose a cryptographic system PLUTUS. The system encrypts files stored on un-trusted servers. PLUTUS associates a symmetric key (lock-box key) with groups of files, which have similar sharing attributes. Each file is encrypted using a file-block key and then afterwards with a lock box key of the file group to which the file belongs. The lock box key is also used to encrypt the file block keys. When sharing a file a user only needs to reveal the lock-box key and then the file can be read by other users. The drawback of the scheme is that it does not ensure fine grained access control, as access is given on the basis of revealing a key.

Another problem with the approach is that performing multiple encryptions at different levels makes this system extremely complicated due to necessity of the management of keys. The scalability of the system is proportional to the total number of file-groups. The

assumption is that the number of users is more compared to number of file groups which makes the scheme relatively efficient. In file-groups, as the number of files grows within a group, the easier it would become to manage the keys from a providers perspective. However, the bigger the group the more coarse grained the access control will become. Therefore for this system the problem of key management is inversely proportional to the granularity of access control.

Ateniese et al. [91], presents an extension of the atomic proxy re-encryption scheme developed by Blaze et al [93]. The fundamental concept of proxy re-encryption is that it allows transforming a cipher text computed by Alice's public key into one that can be decrypted by Bob's private key. Ateniese presents a system in which data owner encrypts the data contents using symmetric keys. These symmetric keys are all encrypted with a master public key. The data owner has the private key that can decrypt these symmetric keys. This private key combined with user's public key is used to generate proxy re-encryption keys. Now the semi-trusted server can convert the cipher text into a form that can be decrypted by a specific user, hence allowing that user access to data. The fundamental problem with this scheme is that a malicious server and one user colluding together can reveal the keys of all the users.

Goh et al. [90] presents a system SiRiUS, which provides end to end security over existing network file systems such as CIFS, NFS or OceanStore. To ensure access control it uses access control list (ACL) and assigns to each file a meta-data file containing the ACL. The meta-data file in SiRiUS incorporates entries that encrypt the file encryption key with corresponding user public key. This enables only the legitimate users to decrypt the file encryption keys and therefore access the file. Users who have the signing keys (private keys) are given access to write as well. The system is heavily reliant on PKI from the scalability perspective for Cloud scenarios the system therefore is not suitable. Furthermore the linking of ACL with the files means that the complexity of the system is proportional to number of revoked users. Every

time user revocation takes place, the meta-data file has to be updated by re-encrypting each entry in the ACL with the new file encryption key. In Cloud computing scenarios files with multiple copies would be stored at different infrastructure providers; if SiRiUS is used then ensuring synchronisation of access control policies would incur immense overhead therefore making the system infeasible. The system does not achieve fine granularity, as decisions about access is made on the basis of provisioning of keys only. For clarity the provider is only responsible for hosting and distributing the data. No processing is performed on the data itself by the provider.

Vimercati et al. [92] presents a solution based on a key derivation method[94]. The fundamental concept is that every user is assigned a secret key and each file is encrypted using a symmetric key. In order to grant access to a user the data owner creates public tokens using the key derivation method. Using the token alongside with the user's secret key, the user is then able to decrypt the file. The benefit of the scheme is that it is not dependent on public key infrastructure (PKI), therefore it should be highly scalable. However, due to the complexity of file creation and user grant revocation makes this scheme un-scalable [95].

Sahai and Waters [96] presented the original idea of attribute based encryption. In this scheme a user's private key is constructed as a set of private key components. Each component represents an attribute of the user's identity. A user trying to read an encrypted file can only decrypt the file if its private key has the correct access control attributes embedded in it. The fundamental problem with the scheme was that it cannot be used in large systems due to lack of expressibility[97]. This scheme however, formed the basis for considerable further research work in attribute based encryption[95][97] [98].

Goyal et al. [97] presented an enhanced version of the Attribute based Encryption (ABE) called as Key Policy Attribute Based Encryption (KP-ABE). The scheme enhanced the expressibility of the ABE scheme by expressing the user's private key as the access formula by using AND, OR, or threshold gates. Data is associated with attributes for which a private

key is derived. The private key is granted to users who have correct attributes. The scheme achieves expressibility to a certain extent but the complexity due to access attributes relating to that of public keys of users still make the scheme extremely complex. Furthermore, there is a considerable computational overhead that is associated with the scheme which makes it unfeasible in the Cloud computing scenario.

With regards to distributed access control the state of the art research is based on the Key Policy Attribute Based Encryption (KP-ABE) scheme[97]. We therefore give some detail on relevant KP-ABE schemes that relate to that scheme.

There are three major operations performed by the scheme, which are encryption, key generation and decryption. To understand how the scheme achieves fine granularity and data confidentiality simultaneously we explain below the functions of encryption, decryption and key generation.

Encryption: To perform encryption the following components are required,

- A set of attributes ***I*** which are associated with secrets
- An access tree ***T*** to which the attributes are mapped. A tree is a Boolean expression which has attributes as its leaf nodes.
- A public key ***PK*** that is used for encryption, this key is known to everyone in the scenario
- Finally a message ***M*** that will be encrypted using the above sub-components

Key Generation: To perform key generation the following components are required,

- The access tree ***T***, that corresponds to the set of attributes ***I***
- The master key ***MK***, this key is extremely critical to the security of the scheme. If this key gets compromised, an attacker can easily impersonate the data owner
- The public key ***PK*** that is known to everyone in the scenario

- Finally the key generation algorithm takes as input the above components and outputs the data consumer's secret key ***SK***

Decryption: To perform decryption the following components are required,

- A Data consumer who performs the decryption needs the ***SK***
- The public key ***PK*** is required
- Finally the algorithm takes cipher text ***E***, using the ***SK***, the process of decryption will be performed. The decryption is successful only when ***SK*** relates to the set of attributes ***I*** in the access structure ***T***

Now the above scheme achieves two very important functionalities. First, by associating the attributes with the secret key of the user mapped over access structure ***T***, it has achieved fine granularity. The second function is that of confidentiality that is achieved by performing encryption of that data. The scheme works well in a scenario where there is an un-trusted server and all the data hosted on it has to be encrypted and also fine granularity has to be achieved. But due to the complexities relating to computational overhead, key distribution and user revocation the scheme does not scale.

4.1.4 RESEARCH GAPS

Research Gap 1 (RG1): In Cloud computing the data is hosted on un-trusted servers; this feature introduces a requirement for confidentiality because the assumption is that the data is hosted outside the control of the data owner. Therefore, a mechanism should be in place that ensures confidentiality and integrity. As the data that is to be hosted on Cloud could be immense, there is also a requirement for fine grained access control, otherwise managing access to the data would not be possible.

In Cloud computing scenario, the requirement is to have a scheme that would provide fine granular access control and data confidentiality. For enterprise to use Cloud computing, they

would need assurance that their data remains confidential and only transparent to authorised users. In case this condition of data confidentiality is not satisfied, it would erode the financial benefit that Cloud computing brings, as it would be risky (loss of customer's trust, fines from regulator, monetary losses) for enterprise to use Cloud computing to host their data.

From the literature review in section 5.2.2, it can be concluded that the schemes of ABE[96] and KP-ABE[97] do achieve fine granularity and data confidentiality. As Cloud computing is going to be used for a huge numbers of users, it would require that the scheme providing confidentiality and fine granularity of access control scale to that required level. The schemes of ABE and KP-ABE do achieve fine granularity but due to the lack scalability, lack of expressibility or computational overhead they are infeasible to be directly used in Cloud computing scenarios. Therefore the requirement is to have fine granularity, data confidentiality and scalability simultaneously.

The problem in the state of the art is linking of fine granularity to the provisioning of confidentiality. In KP-ABE fine granularity is achieved using the access tree structure and as the attributes are linked to a certain identity, only that identity can decrypt data. This makes the scheme very complicated due to the management of keys, development of access tree structures, re-encryption and key generation. Therefore it cannot cater for the scalability requirements of the Cloud computing scenario.

An important research aim is to de-link fine granularity of access control from the data confidentiality and to come up with a scheme that scales to a level required in the Cloud computing scenario. This would require development of a new scheme that takes into account specific access control, confidentiality and scalability requirements of Cloud computing. Scalable fine grained access control would be relatively simple to achieve as the delinking would mean that the mechanism does not have to cater for the confidentiality requirements also.

Research Gap 2 (RG2): Cloud computing should create the illusion that the configuration relating to encryption, decryption, management of security policies and management of keys is done seamlessly. This illusion forms the basis of Cloud computing and it differentiates Cloud computing from other forms of distributed computing.

The challenge is to ensure that security services such as confidentiality, access control and integrity is provided on Cloud just as it would be provided in an enterprise with minimum possible user involvement (both owner and consumer).

None of the systems that we have analysed above provide a solution that performs automation of configuration by which all the security related configuration remains seamless to the users.

Cloud computing has the capacity to scale up and down as per the requirement of the business. This scaling capability is built on top of multiple domains for instance, an enterprise conventionally uses a single domain for its operations. But in Cloud computing it would have the capacity to burst to multiple domains (using broker or directly using infrastructure providers). This capacity has to be built in a way that it remains completely seamless to the enterprise users. The focus of the research gap is to deal with the issues that arise due to bursting. This bursting would include network level, VM level and hypervisor level security configurations in accordance to the security policy of the enterprise. Furthermore, this configuration would include security configuration negotiation (example, negotiation about which algorithms to use and length of the keys), management of keys, encryption, decryption and user revocation.

The systems analysed provide solutions that cater for fine granularity and confidentiality. They do not provide a mechanism to seamlessly perform the security configurations. Consider the example, when a service provider wants to burst to multiple infrastructure providers, at the network level it would be a requirement to ensure confidentiality of data. A potential solution to this problem may be the use of either the Secure Socket Layer (SSL)

or creation of Virtual Private Network between the service provider and infrastructure providers. The usage for instance of the VPN technology would require configuration relating to keys, setting up of VPN servers at the infrastructure providers, setting up of VPN clients and enforcement of network level security policy of the service provider (for instance, if the infrastructure provider is outside of the EU, the bursting should not take place). All of this configuration would require the development of a component that would oversee network level security. This is only one requirement that relates to the security configuration relating to bursting. The goal here is to ensure that all of the security related requirements are fulfilled by a component with minimum possible involvement of the service provider.

For Cloud computing to grow and realize the vision that we have provided in Chapter 2, it is imperative that this research gap is fulfilled. In case this research gap is not fulfilled the risk (time required for configuration, right configuration ensuring all security requirements have been fulfilled) that an enterprise has to take to use Cloud computing may erode its economic benefits. Moreover, the challenge is not only to achieve the goal of automation of configuration but also to achieve it at a cost that enables Cloud computing to remain a lucrative option.

Research Gap 3 (RG3): None of the analysed systems proposes a mechanism by which access control policies that are distributed over multiple infrastructure providers are kept synchronised. In Cloud computing scenarios, the data may reside and pass from numerous platforms like Broker, Service provider and Infrastructure provider (IP). The challenge is to ensure synchronisation of access control policies across these multiple domains. Consider the following example:

Data owner hosts its data on IP1 and IP2, where IP1 provide services to data consumer from Collaboration A and IP2 provide services to data consumer from Collaboration B. Data owner has different access control policies for both Collaboration A and Collaboration B. However, the data owner wants to ensure that if the access control rights of Bob who is a member of

Collaboration A gets revoked, he should not be able to use collaboration B to get access to data. This would mean that both the access control enforcements at Collaboration A and Collaboration B need to be synchronised.

A typical usage of the above example is when Collaboration A and Collaboration B are competing with each other. The data owner does not want to reveal the data that it is giving to Collaboration A, to Collaboration B and vice versa.

Luokai Hu et al.[99] presents a novel approach towards solving the problem of communicating policies between different players within the Cloud computing scenario. The scheme proposed works in a decentralised environment using semantic web based technologies such as XACML. Some elements like the subject, object, action and attribute variables from XACML is used. This approach potentially solves the problem of semantic interoperability and mutual understanding on distributed access control policies when collaboration work is done between organisations. The problem with the scheme is that it does not have an automatic mechanism for resolving conflicts. Furthermore, the level of granularity provided by the scheme is limited as well.

Research Gap 4 (RG4): All the systems that are analysed, user revocation is one of the most cumbersome processes. Every time users get revoked, new keys have to be generated and files have to be re-encrypted. Due to this complexity, all of the systems do not scale to a level that is desirable for Cloud computing scenarios.

Yu et al[95] presented a novel approach in which the problem of user revocation was outsourced to Cloud servers. This was achieved by using PRE [93], which enables the transformation (re-encryption) of cipher text without the Cloud servers knowing anything about the plain text. The problem with this approach is that it introduces a huge amount of computational overhead. This computation overhead itself is outsourced to Cloud servers but this would increase the costs (time, money) of hosting data on the Cloud. Therefore

taking out the economic benefit that Cloud computing brings and making the approach infeasible.

An important research aim is to come up with a scheme that minimises the impact of user revocation on data re-encryption. In the KP-ABE scheme that we have analysed in RG1, attributes are linked with keys in order to achieve fine granularity. The problem with this approach is that every time a user gets his access revoked all the files that he has access need to be re-encrypted. Furthermore, this introduces the issue of computational overhead also making the scheme infeasible.

Consider a scenario where multiple users are having their access revoked in a very short span of time. This would require numerous re-encryption processes starting over the span of a short time. It would mean that if the time taken to re-encrypt data is longer than the time a user gets his access revoked than the scheme is infeasible, as if there are multiple re-encryption processes running then there should be a mechanism to identify which data is encrypted with which key. As KP-ABE does not handle this complexity, it therefore is not feasible for the Cloud computing scenarios.

4.2 DATA LEAKAGE PREVENTION (DLP)

Companies pay a lot of attention to protecting the perimeter of their network by using firewalls and intrusion detection and prevention systems. Though the significance of external security is very important, it is also critical to ensure that data is not lost from inside the enterprise intentionally or unintentionally.

There are products in the market which cater for data leakage prevention in an enterprise like GFI endpoint security [100], Symantec Data loss prevention[101] etc. There are no products which cater specifically for data leakage prevention in the Cloud.

The reason why the problem is different in the Cloud is because leakage within an organization can be controlled by using the above mentioned software. Such software would scan through end points or network points looking for violations of enterprise security policy. Such software cannot be used directly *by customers* in the context of Cloud computing, as the Infrastructure provider is hosting data from various different customers. The infrastructure provider should control the infrastructure and it should apply uniform security policies across many different customers' data.

From the perspective of the Infrastructure provider the problem is even more complicated as, firstly, it would want to prevent the leakage of data from happening. In case it happens, it is first very important to identify where and when did the leakage happened. Secondly, providers want to identify steps or processes by which the violation of the security policy can be traced. Understanding the exact requirements for data leakage prevention in the Cloud is still an open question.

Intrusion detection systems (IDS) monitor an agent's activity and determine whether it is acting as expected or not[102]. IDS can potentially be used to check the behaviour of a user accessing the data. IDS can solve problems relating to the Data leakage by analysing user behaviours and then reporting unusual activity in the system.

The IDS assumes that an adversary has successfully been able to bypass the initial security of the system. For instance the username and password has been compromised by the adversary. For the IDS, the challenge is to figure out how to detect unexpected behaviour using previous patterns of users of the systems. The problem can be further divided into two categories. The first is to find out data that would show expected behaviour. The second one is to develop the software that would detect unexpected behaviour.

There are two categories of IDS, network based or host based. In network based IDS[103], techniques are used to analyse network traffic to figure out anomalies with respect to expected behaviour. This is the new form of IDS compared with the host based intrusion

detection system. Host based IDS[104] consist of four methods for detection: the monitoring of logs, kernel based detection, file system monitoring and connection analysis.

With respect to Data leakage the host based system makes more sense as the main concern here is to protect the data that is stored. The network based IDS can play a complementary role by checking for anomalies in the network. When envisioning the data leakage scenario, one has to consider all the potential output channels (USB drive, CD/DVD burner) associated with a computer. These channels can be exploited by an adversary to move information out of the Cloud. To counter them techniques like profiling of programs on basis of system calls presented in[105] can be used for intrusion detection. Two approaches are generally adapted when dealing with intrusion detection system.

The first approach is that of Misuse detection[106] [107] in which systems are systematically scanned in order to detect previous known behaviours and actions which are deemed malicious or intrusive. These behaviours or patterns are usually recorded over a long period of time relating to a specific system. The second approach is that of Anomaly detection [108][109] in which the assumption is made that any unusual behaviour of the user would be deemed as intrusive. The advantage of Anomaly detection over Misuse detection is that it could even detect novel attacks as it looks for anomalies rather matching previously known patterns.

From the perspective of the Misuse detection technique in DLP, the research question is to determine the behaviour and patterns of users not only from within the organisation but also from customers who are accessing that data from outside. Due to the nature of Cloud computing, data of many different customers would be hosted on the Cloud. The data would be accessed simultaneously by users and customers at the same time. In case the system specific intrusive behaviours are determined then it would enable the application of traditional host based intrusion detection systems on the Cloud platforms.

The approach of Anomaly detection is more novel compared to Misuse detection but its reliance on unusual behaviour is not enough in order to detect intrusion in Cloud. For example a user within an organisation can be performing legitimate tasks while moving data on USB or burning it on DVD/CD. In Anomaly detection, it would not be flagged as intrusive behaviour and therefore the attack would not be detected. This approach performs very well when it comes to attacks that are coming from the network by analysing abnormal behaviour in the network traffic for example running of a port scanner by a machine not in the LAN.

An example of Data leakage not on the Cloud but still relevant is that of Bradley Manning, a US army soldier who was able to move classified data to his personal computer. He then later provided that data to the Wikileaks website[110]. This is an example of how easy it is for employees of an organisation to leak sensitive or classified data. On the Cloud the problem is even bigger as now the employees have access to data of not one but many companies or even at times competitors.

The research question from the perspective of Cloud computing is that there are no known behaviours from the perspective of host based IDS specifically for infrastructure providers. The current state of the art is not specifically tuned to understand the behaviour of internal (employees of the IaaS) or external users (customers) of IaaS. Conducting research in this areas and coming up with user behaviours associated with the scenario would enable the application of traditional host based IDS on Cloud platforms.

Providing the customers with a legal or technical guarantee that the data would not leak is extremely important. Unable to achieve this task would hamper the growth of Cloud platforms and also would enable the bigger names to create their monopoly in the market. This would serve as a barrier for new entrants in market place, as the customers would trust the large providers (Amazon and Google). Furthermore even for large providers, the trust would be limited and the customers would not be able to store with confidence their confidential information on the Cloud platforms.

4.3 HYPERVISOR LEVEL SECURITY

Hypervisor creates the abstraction layer between the guest OS (virtual machines) and the hardware. One can also describe it as a slim version of an operating system with a focus on performance. The Hypervisor is responsible for managing the network level traffic. It redirects the traffic to appropriate virtual machines.

A hypervisor, if compromised, can potentially lead to the compromise of all the virtual machines running on it, therefore inducing a very high risk. Recently some of the vulnerabilities of the hypervisor have been exploited by attackers. An example is the attack on the hypervisor of the Xbox 360 online system by exploiting a buffer overflow to gain hypervisor mode[111]. There are numerous other attacks recorded that have exploited the vulnerabilities of hypervisors[112] [113] [114]. Furthermore, there are numerous rootkit attacks in which slim version of hypervisors infect a virtual machine and then gain control over it. These systems include rootkit Bluepill[115] and VMBR [116].

In [117], Khalid identifies that due to a smaller code base and relatively low complexity the VMs were considered less vulnerable in the past. In fact they were recommended for use to add further security. However, advances in rootkits and other malware have raised the levels of risk associated with VMs. There are two broad categories of security when it comes to VMs. The first one is the security of the VM. In this category research work is conducted to ensure that the VM remains secure. In the second category (VM for security), the research work is conducted to find out how VM can be used to secure systems. The first category is the focus of the work in this thesis.

The integrity of VMs is considered extremely important, as this ensures the security state of the VM. In case the integrity cannot be ensured this would raise a huge question mark on overall security of the VM. VMs in Cloud settings are used by various consumers and it also holds software that is being constantly updated, therefore ensuring the integrity of the VMs is not straight forward. Furthermore, existing security measures relating to VMs focus

primarily on securing the data present within the VMs. There is a huge security requirement to develop controls to ensure confidentiality and integrity of the VMs itself.

Jinpeng Wei et al. [118] presents a framework for managing VMs, this framework provides two benefits, first it reduces the risk relating to unauthorised access by employing access control mechanisms. Secondly, it applies filters to remove unwanted information within the VM. The filters mitigates the risks relating to publishing of unwanted information.

Karger [119] and Sailer[81] presents various security architectures that can be used to secure the hypervisor. These techniques include usage of access control mechanism that allows for the compartmentalisation of the hypervisor. These techniques are focused towards coming up with an architecture that makes the hypervisor secure, whereas none of the architectures identify the security vulnerabilities that would come up when providing API level access to the introspective layer of the hypervisor.

In [119] Karger, explains that there are two approaches to ensure the security of the hypervisor: complete partitioning and isolation of resources of the VMs at the hypervisor level; and sharing of hypervisors, where I/O, memory and other resources are shared between the VMs.

The approach of complete isolation is not useful, as the reason for using Cloud infrastructure is to provide cheap access to computing resources. In this approach only a small number of VMs can be provisioned for a single machine as resources cannot be shared between the VMs. This approach may only be useful for cases where there are large mainframes and the users want complete isolation of VMs. Relatively the security of the hypervisors would be high in this approach because of complete segregation of resources. Giving access to the introspective layer of the hypervisors in this case would be less risky. An adversary who gains control over one partition of the hypervisor would not be able to infect the other partitions due to the segregation.

The second approach (sharing hypervisors) suits the Cloud scenario presented in section 2.4. Securing this approach is a much bigger challenge. Sailer[81][8] presents an architecture (sHype) which secures a Xen hypervisor using mandatory access control. The approach has little performance overhead. The hypervisor sHype is advancement of Xen hypervisor; it controls all VM communication and secures the communication by using formal security policies. The sHype is designed to support security requirements like access control between VMs, isolation of virtual resources and resource control.

sHype does not provide an API by which a programmer can update the security functionalities that are provided. The reason most probably is that, an adversary may exploit the API to infect with malware. As for VMware, the company provides API level access to its hypervisors. This enables developers to deploy customised software to tackle malware, Trojans and viruses. This functionality provided by VMware would enable the development of innovative security products for Cloud platforms.

Another interesting challenge would be to apply the usage control model [13] to secure the access to data by VMs in the hypervisor. In sHype, a mandatory access control (MAC) mechanism is used. The MAC uses a security policy to ensure that the system security goals are accomplished regardless of system user. This is primarily because MAC is designed to be used by operating systems.

MAC depends on objects and subjects to identify access. The control of access remains with the operating system, therefore this setting would not work with Cloud computing as in Cloud there are multiple VMs and multiple hypervisors.

4.4 HYPERVISOR BASED INTRUSION DETECTION SYSTEM (IDS)

Our analysis in the previous sections has identified the introduction of malicious code as one of the most serious security threats. There are numerous techniques for anomaly detection.

Three techniques are widely used for detection of malicious code,namely static analysis, dynamic analysis and manual analysis.

Static analysis[120] [121] [122] analyses the program code without executing the code. Though this technique is exhaustive, a benefit of this technique is that it consumes less time and covers all of the code compared to the dynamic technique. Static analysis primarily depends on tools that provide information about control flow and data flow. This information is primarily obtained from source code. There are products like Checkmarx Code Analysis [123] which provide this functionality to analyst to find vulnerabilities. In static testing there are no runtime request send to web applications and therefore runtime information is not analysed.

Dynamic analysis analyses the code at runtime and only that part of the code is analysed that executes during runtime. Dynamic analysis is normally performed on virtual machines, as using standalone machines would mean having to reinstall the operating system after every test. The advantage of this technique is that it is non-exhaustive and only the code executed is analysed[124]. There are products such as Acunetix Web Vulnerability Scanner [125] which enables an analyst to perform vulnerability assessment on web portals. Major vulnerabilities that are identified by this technique are cross site scripting, SQL injection etc.

The two techniques though are completely different but can be used in conjunction to complement each other [126]. Moreover, manual testing is another technique that is used by analyst to identify vulnerabilities. Manual testing is done by experts to compliment the automated tools that are used for penetration testing. Kali Linux[127] provide tools that are used by experts to perform manual testing.

The static and the dynamic analysis are both performed on code that is either running or present on a machine or VM that the analyst has control on. The analyst performs the analysis using the widely available tools and techniques relating to the static or dynamic analysis. This approach and tools cannot be used directly in Cloud computing scenario where

it would be desirable from the hypervisor level which is owned by the IaaS to run analysis on VM which is owned by the customer. Although the security responsibility of the VM remains with the customer, from the IaaS perspective it would be highly desirable to ensure and provide extra level of security to the VM. For instance a VM infected with malware may not be able to recover to its un-infected state. The hypervisor should in this case perform an analysis to first identify a malware and then counter it.

A research challenge is to come up with an analysis technique by which the malware patterns in the virtual machines can be analysed from the hypervisor level. In case a malware infects a virtual machine, then the malware has the same level of access rights as the guest OS. In case of the hypervisor, if the VM gets infected than the hypervisor can potentially detect and delete the malware without infecting itself.

This research can lead to a novice technique where patterns on the VM can be analysed from the hypervisor level. The major benefit of the technique would be that when performing the analysis the hypervisor will not get infected itself. Furthermore, it would enable IaaS providers to provide the customers with an added layer of security for their VMs.

A virtual machine monitor (VMM) [128][92] is a light software that runs directly on the hardware of a machine. The VMM virtualises all hardware enabling the virtual machines to transparently consume the resources of the physical machine. Hypervisors are closely linked with VMM [129] and often the term is interchangeably used.

In [130] Garfinkel describes a system (Livewire) by which the monitoring of VMs can be done by using VMM. Furthermore it also enables the limited detection of malware at VM. The system is based on the technique Virtual machine introspection (VMI). VMI is used to inspect the activities performed by a VM, by using the knowledge of OS level semantics. This knowledge enables the interpretation of events on the VM from the VMM level. Livewire is developed on top of the closed source hypervisor VMware[131].

Intrusion sensing and introspection system (ISIS) [132] uses a different approach compare to Livewire, as it is developed as a component that can be added to a hypervisor system. In Livewire the hypervisor itself was extended. ISIS is built on a User mode Linux system in separate kernel address space mode (SKAS). ISIS use a system call Ptrace that enables one process to monitor and control another process. Ptrace also provides the ISIS the capability to modify kernel code at runtime.

Livewire puts hooks on the VMM. ISIS puts hooks on the guest operating system kernel. However an attacker who has access to the kernel code can remove the hooks placed by ISIS. Both ISIS and Livewire are susceptible to attacks where an attacker modifies the code of the kernel without triggering an intrusion sensor [132].

Both these approaches focus on monitoring and detection of intrusive patterns. They do not provide the capabilities of intrusion prevention, spreading and deletion of malware. A Prevention capability would require the identification of malware at the network level before it infects the VM. Spreading of the malware is a complicated problem, as it requires a mechanism by which the hypervisor stops the spreading of malware from one VM to the other, by blocking the network traffic.

From our use cases (explained in Chapter 2), we deduce that the outsourcing of IT infrastructure to Cloud platforms would mean that different customers would be using VM running on the same hypervisor. From the perspective of the infrastructure provider it is very important to ensure that malware is detected well in time and that if one VM gets infected the malware does not spread to other VMs.

We know that an attacker can modify the kernel code of the guest operating system which would disable systems such as ISIS and Livewire before they detect the attacker. The research challenge is to ensure that an attacker who has access to one VM does not spread malware to the other VMs on the hypervisor. Another challenge would be the understanding

of attack patterns. Knowledge of the attack patterns would increase the accuracy of the detection, therefore decreasing the number of infections.

4.5 RISK MANAGEMENT

The hosting of data on Cloud platforms and using computing services from the Cloud brings new challenges. The risks that a Cloud consumer is exposing itself to is different than scenarios where data and processing requirements are consumed locally.

Furthermore, for the Cloud provider the requirement of provisioning appropriate resources at a certain point in time is also very important. The risk of not having enough resources at the disposal of the Cloud provider can potentially cause major disruptions for its customers. This would mean loss of reputation and potential loss of business as well. In [133] a mechanism is developed using Bernouli's theorem which is primarily used in the financial sector to predict liquidity (resource bank) risk. For Cloud computing the prediction about the resource bank becomes very important. The model proposed is quite interesting from the perspective that a Cloud provider can potentially predict the resource related risk that it is getting into and then can plan appropriately.

In [134] a risk analysis is done for the hybrid Cloud model. The use of public Cloud is cheap but it comes with its security risks whereas the use of private Cloud is expensive, however, the security risks are much less. Therefore, a lot of the companies are opting for hybrid Cloud. In a hybrid Cloud setting, the companies would normally opt for private Cloud for critical data or services. At the same time they have the option to burst to public Cloud if the requirement exceeds the available supply. The paper analyses 21 different risks for the hybrid setting and then propose countermeasures. The paper however does not quantitatively measure the effectiveness of proposed countermeasures.

From the analysis of the state of the art we deduce that there are open questions to be addressed on business driven risk associated with Cloud computing. There is requirement to

develop a framework that would take into account the security risk associated with Cloud computing platforms. This framework can enable a consumer to trust the Cloud provider more and therefore can lead to better adoption of Cloud services.

4.5.1 RESEARCH GAP

Research Gap 5 (RG5): The challenge of mitigating risk associated with hosting data on the Cloud or using Cloud services in general is a major cause of concern for enterprises moving to Cloud. To mitigate such risks there is a need to undertake research for the development of a risk framework for Cloud computing from a security perspective. This framework would enable the mitigation of risks associated with Cloud platforms.

The requirement is there to develop mechanism through which the risk framework can gather information about different security controls such as firewall configuration, intrusion preventions system logs, traffic analysis etc. and then relate them with different risks such as hacking, data theft, denial of service etc.

Furthermore, the requirement is also there to classify different threat categories. In order to raise the correct level of alert it is pertinent to first rate different threats and classify them appropriately. For the classification of risks, one needs to either develop a new threat analysis mechanism for Cloud computing or use the existing ones and tune them with respect to Cloud scenarios.

Research also needs to be conducted in the area of mathematically calculating a risk, this calculation would then lead to raising the alert or not. For instance, if the firewall logs show that there might be a contamination, should this be treated as an alert, has the threshold for raising the alert breached or not. In a lot of other security software, so many alerts are raised that it is not humanly possible for the system administrator to study each of them and then act appropriately.

Therefore the next challenge would be to take actions based on the level of alerts that have been raised. In case the threat level has been raised and a risk is about to mitigate like denial of service. The risk framework could potentially switch part of the traffic to backup server to maintain quality of service at a decent level.

The researcher has contributed to the ENISA paper on Cloud Security titled as 'Cloud Computing: Benefits, Risks and Recommendations for Information Security'[70]. Moreover he has also participated in developing security controls for different certification of Cloud Security Alliance[135]. This research work has enabled the researcher to develop the 'Cloud Security Framework' which is detailed in Chapter 5.

4.6 RESEARCH AGENDA

In this section tools, technologies and models are discussed that will be used to fill the gaps introduced previously.

To fill the gap introduced in RG1, we survey secret sharing schemes such as Shamir's secret sharing [136] and decide which ones are suitable. The fundamental requirement is that the scheme should be able to scale and offer fine granularity.

In order to achieve the aim of automation of security services in Cloud computing scenarios as mentioned in RG2, there is a need for the development of a prototype that would simulate this automation. In order to achieve this task, work has started on understanding the current state of the art solutions relating to access control. One of the suitable solutions would be selected and it would be extended. In case none of the solutions is deemed useful to achieve our task then the development of a new tool that would simulate the automation would take place.

For RG3, the challenge of synchronisation of access control policies is a new one and there is hardly any research work that has been conducted in this area. We therefore propose to

extend the scheme that we develop for achieving fine granularity and scalability in order to achieve concurrency.

For RG4 all the systems that are analysed in section 5.2.2 that achieve fine granularity in distributed access control scenarios are built upon attribute based encryption (ABE). The problem is that the ABE inherently induces cumbersome problems of user revocation and key management. Therefore it is proposed to take a step backwards and to approach the problem of fine granularity and user revocation for Cloud computing scenarios with a completely new approach.

For RG5, the requirement is to first undertake a detailed threat analysis for Cloud platforms based on different Cloud scenarios such as the ones presented in Chapter 2. Once the threats relating to Cloud computing have been identified, it is then pertinent to develop a framework than can accurately raise alerts for threats that are going to or are materialising.

4.6.1 RESEARCH OUTCOMES

What follows are the proposed research outcomes for this thesis:

1. Development of a scheme that would enable the enforcement of access control security policies on Cloud platforms ensuring scalability and confidentiality. This task is relevant to the research gaps RG1 and RG4.
2. To deploy the tool for a project (OPTIMIS or Internal BT project) to perform security evaluation of the scheme with respect to the relevant case study. This task is relevant to the research gaps RG1 and RG4.
3. Design and development of a security risk framework to analyse the risks associated with the Cloud computing platform. This task relates to RG5.
4. Development of a security risk framework tool that would ensure the risk analysis could be performed for Cloud platforms. This task relates to RG5.

CHAPTER 5

A SECURITY RISK FRAMEWORK FOR CLOUD COMPUTING

This Chapter puts forwards a Security Risk Framework relating to the security challenges posed by the management of risk for Cloud computing environments. Furthermore, for the Security Risk Framework a detailed threat analysis of Cloud computing across its different deployment scenarios (private, bursting, federation or multi-Clouds) was undertaken. This Chapter fills the research gap 5 presented in Chapter 4.

We also present a risk inventory which documents the security threats identified in terms of availability, integrity and confidentiality for Cloud infrastructures in detail for future security

risks. We also propose a methodology for performing security risk assessment for Cloud computing architectures presenting some of the results.

5.1 INTRODUCTION

In Chapter 4, we have identified a research gap relating to the development of a Cloud specific risk framework. In this Chapter we are addressing this research gap by proposing a Cloud based risk framework.

Section 5.2 explains the motivation and presents the background for the security issues that need to be addressed in Clouds from the perspective of risk management. Section 5.3 puts forward the methodology adopted for developing the risk framework. Section 5.4 provides a systematic approach to threat analysis based on standard threats for distributed systems, adapted for Cloud computing. Initially the Threat and Vulnerability Assessment tool (T&VA) was used to identify threats and vulnerabilities in different Cloud scenarios. The data used to perform this analysis came from the Information Security Forum (ISF) and public data specific to Cloud computing security. Secondly the CORAS risk modelling methodology [9]–[11] coupled with Information Risk Analysis Methodology (IRAM) to complete the risk assessment. Section 5.5 puts forward the high level analysis of each threat for the risk framework, and sections 5.6 and 5.7 explain the part of the risk framework where risk is evaluated and treated respectively.

This research is exploited into a risk model for security and presented in Section 5.8 with an evaluation of the suggested methodology. The results have been based on the implementation work carried out in an EU-project OPTIMIS presenting analyses across different deployment scenarios.

Finally Section 5.9 presents the conclusions of the risk modelling methodology and future research directions to adopt using it.

5.2 PROBLEM STATEMENT AND MOTIVATION

Computer and information security are concerned with ensuring the availability, integrity and confidentiality of information. Availability is concerned with the information being accessible when needed, whereas integrity refers to not allowing data to be modified without being undetected. Confidentiality is concerned with the disclosure of data to unauthorized personnel. Each of these aspects covers an integral part of security aspects of the infrastructure. In Cloud computing, security is one of the highest concerns as it can make or break deals by either convincing organizations to use or deferring its use on security concerns. Others [137] have identified policies and control, knowledge and performance management by using risk, audits, SLA monitoring and protection policies for Clouds. Threat analysis is a preliminary investigation to identify threats relating to cloud computing scenarios. Figure 6 depicts an example of the process relating to threat identification and protection. The protection techniques could be the deployment of a Firewall or IDS/IPS implementation etc.

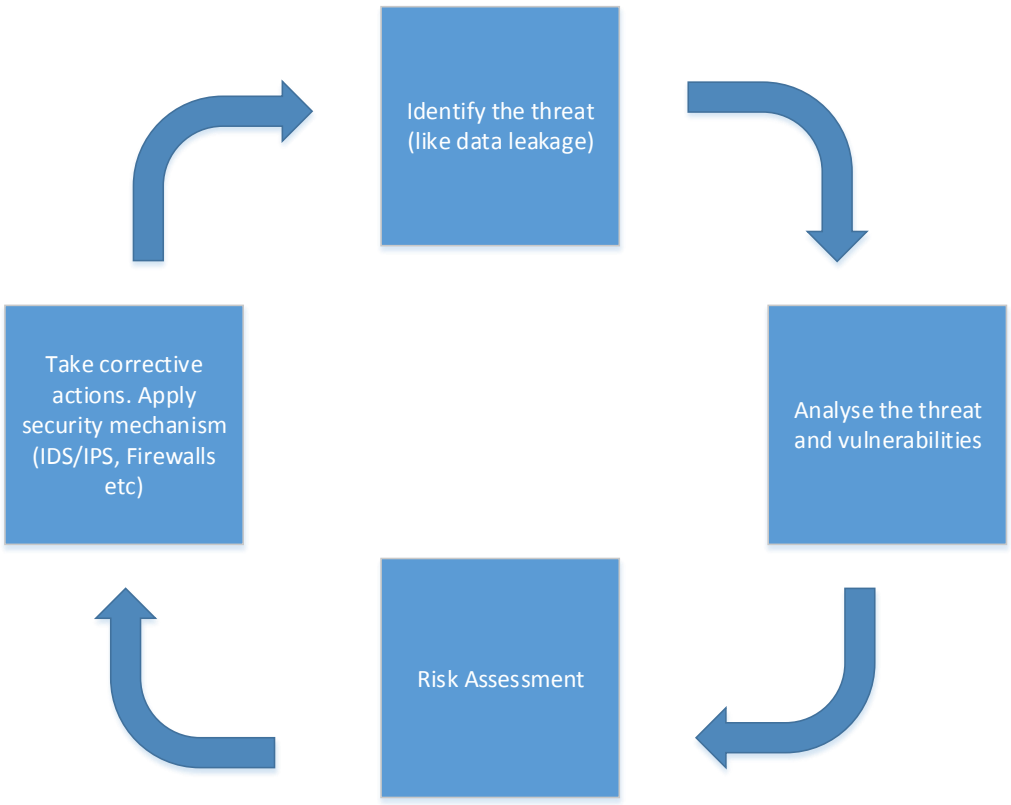

Figure 6: Process for Security Threat Analysis

The different Cloud deployment scenarios raise different kinds of threats depending on how the service executes on the infrastructures. These have been depicted in Figure 7.

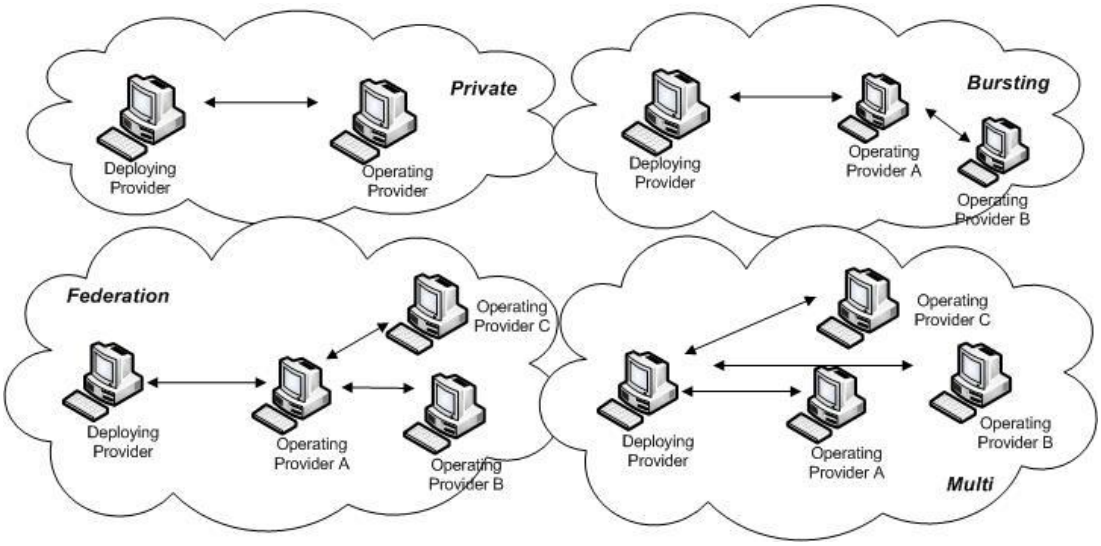

Figure 7: Cloud Scenarios. (Private involves one deploying and one operating provider, Bursting - the operation provider can burst to another provider, Federation - a team of providers work together, Multi - the service can be deployed on a number of providers, acts as a broker)

In the Cloud scenarios we present, there are two primary providers: operating provider and deploying provider. The deploying provider is responsible for deploying the Cloud services like deploying the hypervisor, operating systems, virtualisation and physical server deployment.

The operating providers have their Cloud implementations deployed in the operational phase. The deploying provider would normally consume the operational providers in different scenarios like multi Cloud, federation, bursting and broker.

5.3 METHODOLOGY

This section presents the stages of risk relating to the operational and deployment phases of Cloud scenarios. The risks relating to security from operational and deployment perspective are different.

In the operational stage, the risks largely relate to security vulnerabilities present within the toolkit API's and other portals which are exposed to the external parties. To mitigate this risk, dynamic and manual testing on the API's and end points which are vulnerable should be undertaken.

Risk analysis can be considered at various phases of interactions in Clouds (Figure 8). Each provider involved in the Cloud will have security concerns from their own point of view towards the others in terms of trust, service risks or legal issues. They might consider the risk of working with other providers or may have specific security demands that need to be honoured. These assessments also depend on the Cloud deployment scenarios - private, public or hybrid.

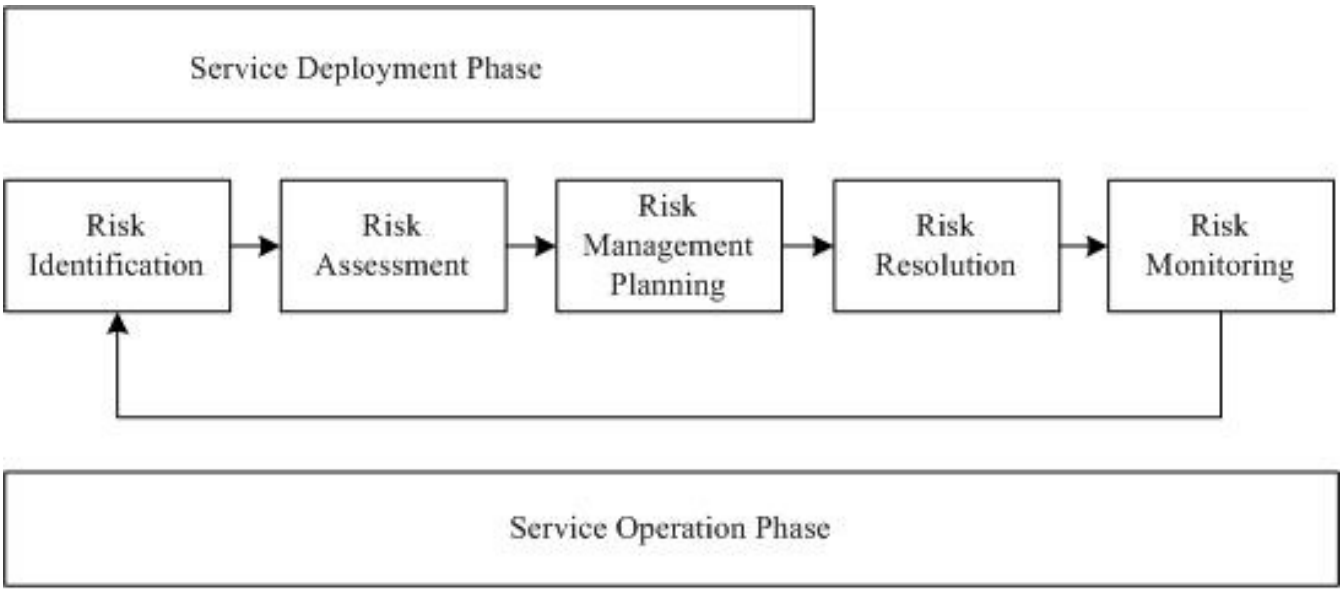

Figure 8: Risk Assessment Lifecycle during Service Deployment/Operation[138]

These concerns can also be refined depending on the stage of the Cloud lifecycle – deployment or operation. Risk needs to be assessed at service deployment stage for initial placement of services on Cloud providers, and the service operation, where Cloud resources and data are managed by the Cloud provider to fulfil the Service Level Objectives. During

deployment and operation stages, risk needs to be constantly monitored in order to prevent any additional costs to be incurred to the end-users and Cloud providers.

The risk analysis methodology was developed as part of the OPTIMIS project, it was then used by different researchers to identify risks and mitigate them. However, the focus of this work remains primarily on information security. The reason why the methodology is mentioned here is to explain the process that was adopted to identify, assess, manage, plan, resolve and monitor risks.

The monitoring part primarily includes running vulnerability assessment exercises on deployed and operational infrastructure at regular intervals to ensure that threats are mitigated.

A number of stages have been identified for performing a complete risk assessment on Clouds by considering core risk assessment approaches as explained below:

5.3.1 HIGH LEVEL ANALYSIS OF THE SYSTEM

An initial high-level analysis of the deployment scenarios helps identify the actions and assets involved at the different stages in the Cloud. Generally security needs to be assessed before deployment of the service to check for security concerns of the other provider or if service level agreements (SLAs) demand certain security aspects to be met. During the operation, security concerns are monitored while the service is executing.

5.3.2 IDENTIFYING THE ASSETS INVOLVED

There are various assets involved either at the deployment or operation stage such as the SLA or customer data. These can be monitored in relation to the specific threats in the environment. To identify the relevant threats with respect to the scenarios of Cloud computing, we must undertake a comprehensive threat assessment.

5.4 THREAT ASSESSMENT

In this section a detailed threat assessment is performed for the Cloud computing scenarios.

5.4.1 METHODOLOGY

Threat modelling is a systemic approach by which threats for and vulnerabilities of a system are identified [139]**.** We have chosen the Information risk analysis methodology (IRAM) coupled to the Threat and vulnerability assessment tool (T&VA)[12] to carry out threat modelling, because these already contains a threat model for distributed systems and software in general. We refined this model to adapt it to Cloud applications.

In this approach we start with a classification of threats. We are using two sources of information for gathering information on attacks on IT systems. The first one is the information security forum which provides data about attacks on IT systems. They also provide the frequency of attacks and the year when the attacks took place. The second source is public data on attacks on the Cloud platforms Amazon EC2 and Google Apps Engine [28]. This contains articles, white papers and research articles.

We combine these threat lists with additional threats identified within the scenario and the use cases that we presented in the previous section.

For vulnerability assessment our approach is the same as for threat assessment. The only difference is that, because the use cases are not currently deployed, some information may be imprecise. This forces us to make a few assumptions about the vulnerabilities. We indicate those in the text.

The T&VA tool provides a standard list of threats relating to IT systems, we have taken the threats from this list relevant to the use cases that we are working on. We have further added threats which are introduced due to the differences between Cloud computing and other

forms of distributed computing. The threats are *Data Leakage, Usage Control and Hypervisor level attacks*. We classify the threats into the following six categories: *External attacks, Theft, System specific threats and abuse, Service interruption, Human error and System malfunction.* These threats have been shortlisted from the threat list provided by the Information Security Forum.

5.4.2 EXTERNAL ATTACKS

We have identified twelve threats in this category starting from the 'Carrying out of Dos (Denial of Service) attack'. Amazon's public Cloud offering EC2 came under DoS attack[140] that left part of its infrastructure unavailable for almost 16 hours. For the scenario that we are using the deployment of Cloud would be either publically available or use public infrastructure. This threat is clearly of high relevance to the cases that we have presented.

The second threat that we have identified is that of 'Hacking'. Using audits such as that of SAS type 2 audit [44] and ISO 27001 [141] most of the current deployments introduce security features in the system that makes them harder to hack. Similarly we were not able to find any Cloud specific attacks with respect to hacking. But due to the nature of Cloud computing as a distributed system it remains susceptible to hack attacks.

The third threat that we have identified is that of 'Undertaking malicious probes or scans'. An attacker can use the publically available deployments of Cloud platforms to run scans and exploit network level vulnerabilities. The ISF database shows a three-fold increase in this kind of attack between 2005 and 2009.

'Cracking password', 'Cracking keys' and 'Spoofing user identities' threats remain relevant to the use cases as these threats can lead to the loss of confidentiality and integrity. Though the likelihood of these threats becoming vulnerabilities remain low according to the ISF provided data. Spoofing user identities can potentially be a challenge in the use cases that we have described especially the Cloud broker one (use case2); as multiple enterprises using

the broker to access infrastructure provider services requires the development of technical and legal safe guard for the protection of identities. The extension of an enterprise's identity management system to a broker and to the infrastructure provider can open up a lot of security related issues. We have already witnessed an identification of a severe security flaw in the Google Apps Engine which can be potentially exploited by a dishonest service provider (broker) to impersonate a user [62].

'Modifying network traffic' and 'Eavesdropping' threats are classified by the ISF as very low level. We believe that due to the distributed nature of Cloud computing and multi layered deployment model, it would be easier for an adversary to exploit these threats.

'Distributing computer viruses', 'Introducing Trojan horses' and 'Introducing malicious code' threats lead to the infection of Cloud platforms. In [142] attack services are defined in which Cloud platforms can be infected with malicious code. In [115] a malware is developed by which a very thin level hypervisor infects a virtual machine. This hypervisor is called Blue Pill and it can then be used to control the VM. The attackers claim that the malicious code remains 100% invisible to the VM and it does not exploit any vulnerability of the operating system it infects.

In [76], we have witnessed that Amazon EC2 Cloud is used by an adversary to distribute spam which led to the banning of EC2 related IP addresses by anti-spam groups. We have identified 'Distributing Spam' as one threat and according to the ISF database the SPAM related attacks are on the rise. In both the Cloud broker and Cloud burst use cases, an enterprise acting as an adversary can use the infrastructure provider's resources to launch SPAM attacks.

5.4.3 THEFT

Cloud computing supports a notion of multi-tenant architecture, in which multiple users consumes the same computing resources. This can lead to the theft of data by an adversary. The threat 'Theft of business information' by either an internal employee or by a competitor

is very relevant to the Cloud brokerage use case as multiple enterprises will be using the broker and infrastructure provider to host their enterprise related data. Google in its security data sheet[143] mentions that only references to the data are deleted rather than data itself. A potential adversary can use advance data recovery tool to recover data owned by other customers.

The next threat that we have identified is the 'Theft of computer equipment'. The likely hood of this threat being exploited is low from the data provided by ISF database. Companies like Salesforce [57][56] employ high end physical security measures to secure their data [57]. For our use cases is extremely useful to understand the physical security measure deployed by the large vendors and therefore lowering the risk.

5.4.4 SYSTEM MALFUNCTION

The ISF database classifies the threat of 'Malfunction of software' as high. Most of the Cloud software like CRM for Salesforce or Cloud APIs for EC2 are extensively used. Therefore a bug in anyone of them can have a very adverse consequence. In our use cases we will consume APIs provided by these infrastructure providers therefore we should take into account necessary control to mitigate this risk.

Malfunction is a mechanism, the threat that we are identifying in this section is the leakage of information due to system malfunction.

The next threat in this category is that of 'Malfunction of computer network equipment'. The likelihood of this threat materialising is high but as our use cases will deploy multiple backups of computing resources the impact will be low. ISF classifies this threat as one of most frequent.

5.4.5 SERVICE INTERRUPTION

One of the Amazon data centre was hit by lighting that led to suspension of part of their service [90]. 'Natural disaster' meaning earthquake, fires, extreme weather and flooding that can lead to the interruption of service. In both our use cases the dependence on infrastructure providers is high and therefore the interruption of service can take place.

'System overload' means excessive system activity leading to the degradation of performance or failure. We have witnessed in[140] that the website Bitbucket that was hosted on the EC2 become unavailable due to a DoS attack. Although Cloud computing offers theoretically unlimited amount of computing resources but it still depends upon how you have configured your website and which availability zone the website resides in. On the contrary Wikileaks [144] used EC2 platform to host their website to protect against DoS attacks, as they were willing to pay for a high end package which means that even if their website is attacked it will still not become unavailable. In use case1 interruption in service can be picked up quickly as the enterprise is interacting directly with the infrastructure providers. Use case2 would be difficult as an interruption in service can be either at the infrastructure provider or at the broker end. The problem becomes more complicated, as it is difficult for an infrastructure provider to determine whether there is genuine peak in demand or a Dos attack as they both may create similar patterns.

5.4.6 HUMAN ERROR

'User error' threat means mistakes made by users of the system. Infrastructure providers like EC2 have designed their system to be automated and there is hardly any human intervention, however they have no control over how users from enterprise or broker may use the system. ISF classifies this threat as high, although in the past few years the frequency of errors have come down but it still remains a high level threat. Google Apps Engine in its

SLA promises 99.9 percent for data availability but does not take responsibility for data loss, as most of it is because of human error [145]. Furthermore, the IT policy compliance group suggests that 75 percent of all data lost is due to user error [146]. 'IT network staff errors' means mistakes made by users who are responsible for maintaining and operating the system. ISF classifies this threat as very high.

5.4.7 SYSTEM SPECIFIC THREAT TYPES

'Data Leakage' is defined as the unauthorised transmission of data (or information) from within an organisation to an external destination or recipient. This may be in electronic form or by means of a physical method [147]. This threat becomes more critical in Cloud environment as now enterprises who are hosting their data on a Cloud have no control over the infrastructure provider's infrastructure. Therefore previously the threat was that it could be stolen by an internal employee but now it could be an internal employee or staff of the infrastructure provider. Even worse it could potentially be a competitor using the same infrastructure provider who is able to steal the data by using advance data recovery tools. For both our use cases this threat is high but it is more critical in the use case2 as data or identities can potentially be stolen at the broker level also.

'Usage control' is a generalization of access control to cover obligation (requirements that have to fulfilled by the subject for allowing access) and conditions (subject and object independent environmental requirements) [13]. In Cloud specific environment where data from multiple enterprises may reside in the same Data centre, it is pertinent to build controls that would ensure that not only access to data is controlled but also that environmental factors are taken into account before allowing that access. Data will also be accessed by applications and regulating that access may require fulfilment of requirements from the side of the subject. A requirement from the enterprise can be that they want complete isolation of their data from any access from roles or applications that are being used by other enterprises. We classify this threat as medium as we have witnessed an attack in which

Google Docs marked as collaborative data from some users which led to disclosure and amendment of that data[148].

'Hypervisor level attacks' enable an adversary to exploit vulnerability at the virtualisation layer that is running underneath the virtual machines. There are numerous attacks that have been recorded at the hypervisor level ranging from the injection of malware to the hijacking of a VM by a thin undetectable hypervisor [149][115][150][151]. Therefore we have classified this threat as high.

5.4.8 VULNERABILITY ASSESSMENT

The vulnerability assessment is based on four methods of analysis which are; the control analysis, environmental analysis, system analysis and technical analysis. For the technical analysis, we didn't have the data available to conduct such an analysis, as our system is not currently deployed. The tool is organised in a manner that asks the user questions with regards to the deployment of the system, since our system is not deployed therefore we have answered the questions by using publically available data about similar deployments and by also making assumptions.

For control analysis, we have covered a wide range of questions ranging from the creation of comprehensive security policy to the security related training of staff. There is a specific set of questions that relate to each vulnerability. Refer to Appendix B for the threat and vulnerability assessment report.

Environmental factors affect the vulnerabilities of a system, these external factors includes economic growth, legislation, regulation and political stability etc. Compliance with regulation is one of the major environmental factors that we have identified. To prevent fraudulent activities, most of the developed nations have passed their own laws and regulations to protect data and ensure privacy. These laws are not specific to Cloud computing but they influence Cloud computing in many different ways. These laws vary from

country to country. For example in the United States the 'sectoral' approach to data protection legislation is preferred while EU follows the overarching government regulation[152][153]. We have also taken into account economic factor, such as the growth of sales of the Cloud related products. We have classified the vulnerability of the Cloud systems to be low if these systems are present in countries which have strict compliance laws in place.

Vulnerabilities are not only affected by the weakness of controls but also ISF information security status survey suggest that characters such as complexity, number of users accessing the system and connectivity to the internet can increase the likelihood of a system experiencing an incident[12]. Most of the Cloud computing infrastructure use the internet to provide services to its customers and infrastructure providers such as Amazon experience high level of connectivity to their systems therefore increasing the chances of vulnerability being exploited. The complexity depends upon use case, for instance the level of complexity will be high in usecase 2 involves more actors and several levels of interaction. As for use case1 the interaction would mainly be between the enterprise and the infrastructure providers.

For the overall ratings we have taken where possible the average of the results that came out from the analyses.

5.4.9 RESULTS OF THREAT ASSESSMENT

The final stage in the threat assessment process is to undertake an information risk rating. The information risk rating requires input from the analyst about the priorities that should be set for the assessment. For instance, threats relating to confidentiality would be higher than threats related with integrity. This information risk rating would then give scores to threats depending upon this input.

To create the information risk rating, we classify the threats of confidentiality as high, availability as medium and integrity as low. We classified confidentiality as high because loss of confidentiality can have a severe effect on trust and image of the provider. Moreover loss of confidentiality, can convert low threats like theft of business information is to very high because loss of unencrypted data is a more severe risk than loss of encrypted data.

Loss of availability is relatively classified as medium compared to loss of confidentiality. The reason being that the impact was not as severe, for instance the Bitbucket website experienced a DoS attack on EC2 infrastructure [85] that led to suspension of their service. But they kept using EC2 after that, they have also reported further attacks on their website but are still using EC2[154].

One of the reasons could be that, the kind of computation power provided by infrastructure providers like Amazon requires a lot of investment. Enterprises are better off using infrastructure provider's resources rather than deploy their own. For instance, Wikileaks recently used the Amazon EC2 Cloud to host their website, when it was constantly attacked using DoS [155]**.** Later under USA government pressure the EC2 platform stopped hosting the website. This shows that if properly configured and enough resources are provisioned by an infrastructure provider then it is difficult to bring down a website using DoS attacks.

We classified integrity as low because relative to confidentiality and availability the impact is much lower. Loss of integrity can be because of many reasons like software error, user error, and equipment failure and also due to an adversary changing the data. From the recorded attacks on Cloud platforms, we have only witnessed a reported incident relating to integrity. It was not very clear, what the actual reason of the integrity checksum failures was in the attack[156]. Furthermore, the VM uses in the Cloud scenario are started, restarted and redeployed on different infrastructures. Therefore, it further enhances the chances of them losing integrity due to errors.

Table 5 lists the various threats identified along with the stage of the Cloud lifecycle these threats may be active. The table also includes the classification of the threats in confidentiality, availability and integrity using the information risk rating.

Table 5 has 5 columns, Column 1 provides an indication of the category of threat under consideration. These threat categories are obtained from the IRAM tool. Column 2 indicates the specific threat under consideration. This column also mentions the words AIC where A stands for availability, I for Integrity and C for confidentiality. Wherever the abbreviations A is mentioned it means that the threat only relates to availability. Same is true for integrity and confidentiality. The column 3 indicates the stage of the Cloud deployment whether it is operation or deployment stage. The column 4 mentions the asset involved like 'customer data'.

The Cloud deployment scenario is column 5 relates to different scenarios like bursting, federation, multi Cloud etc. Column 6 mentions the priority that is linked with the asset. Now this priority has been declared in section 5.4.9. The assets relating to confidentiality are high priority (4 or 5), assets relating to availability are medium priority (3 or 4) and assets relating to integrity would have low priority (1 or 2). Column 5 is relating to likelihood which is the possibility of risk materialising. The likelihood rating has been added by the researcher himself using his own knowledge of the domain. No metrics exist that provide likelihood ratings of Cloud computing scenarios.

There maybe cases were the priority of an asset would be high because it impacts confidentiality but the likelihood of a threat actually materialising would be low. The risks which have high priority and high likelihood are the one which have the highest impact.

The threat numbers (T1, T2, T3,…) in the table are those used by the IRAM tool. Some of the threats recognised by the IRAM tool are not relevant to Cloud computing scenarios and have been omitted from our considerations. However, IRAM's numbering convention has been maintained.

Threat Category	**Threats (threat id) {Threat classification – Availability (A) Confidentiality (C) Integrity (I)}**	**Stage of Cloud (Deployment/Operation)**	**Assets involved**	**Cloud Deployment Scenario**	**Priority (1 is low, 5 is high)**	**Likelihood (1 is low, 5 is high)**
External attacks	Carrying out of Dos (Denial of Service) attack (T1) {A}	Operation	Customer data, infrastructure of the provider	All	4	3
	Hacking (T2) {I,C}	Operation	Customer data or service	All	3	1
	Undertaking malicious probes or scans (T3) {I,C}	Operation	Hypervisor code	All	4	2
	Cracking password (T4) {A,I,C}	Operation	Customer data or service	All	3	1
	Cracking keys (T5) {A,I,C}	Operation	Customer data or service	All	3	1
	Spoofing user identities (T8) (A,C) {A,C}	Operation	Customer data or service, all services	All	3	1
	Modifying network traffic (T9) {I}	Operation	Software, connections, service (runtime)	All	2	2

	Eavesdropping (T10) {I,C}	Operation	Software, connections, service (runtime)	All	2	1
	Distributing computer viruses (T11) {I}	Operation	Software, connections, service	All	3	1
	Introducing Trojan horses (T12) {I}	Operation	Software, connections, service	All	3	1
	Introducing malicious code (T13) {C}	Deployment and Operation	Software, connections, service	All	3	3
	Distributing Spam (T15) {A}	Deployment and Operation	Mailing lists	All	1	4
Theft	Gaining unauthorized access to systems or networks (T16) {A,I,C}	Operation	Customer data or service	All	5	4
	Theft of business information (T27) {A,C}	Operation	Customer data	All	4	2
	Theft of computer equipment (T29) {A,C}	Operation	Customer data	All	1	2

System malfunction	Malfunction of software (T34) {I}	Operation	Toolkit, all services	All	1	4
	Malfunction of computer network equipment (T35) {I}	Operation	Toolkit, all services	All	1	5
Service interruption	Natural disaster (T40) {I}	Deployment/Operation	Customer data	All	1	3
	System overload (T41) {A,C}	Operation	Customer data,	All	4	3
Human error	User error (T42) {C}	Deployment/Operation	Data	All	5	3
System specific threats and abuse	Data Leakage (T50) {I,C}	Operation	Data	All	5	3
	Usage control (T51)	Operation		All		
	Hypervisor level attacks(T52) {A}	Operation	Data	All	3	2
	Data ownership (T53) {I}	Deployment	Data	All		2
	Data exit rights (T54) {I,C}	Deployment	Data, SLA	All	4	3
	Isolation of tenant application (T55) {I,C}	Deployment and Operation	Data	All	5	2
	data encryptions (T56) {A,I,C}	Operation	Data	All	5	3

	Data segregation (T57) {A,I}	Operation	Data, programs	All	4	2
	Tracking and reporting service effectiveness (T58) {A,I}	Operation	Data, Hosted VMs	All	5	3
	Compliance with laws and regulations (T59) {A,I}	Deployment and Operation	Data	All	3	2
	Use of validated products meeting standards (T60) {A,I}	Operation	Data	All	3	3
	Guest virtual machines (T61) {A,I}	Operation	Data	All	1	3

Table 5: Threats Identified in the Various Use Cases and their Details

5.5 HIGH-LEVEL ANALYSIS OF EACH THREAT

Each of the threats can be further analysed in terms of who causes them and the incidents leading up to them, which can then be prioritized depending on this information. This also helps measure the impact of the security risk on the service and the providers. Figure 9 depicts an example of the hacking threat and its related asset and vulnerabilities.

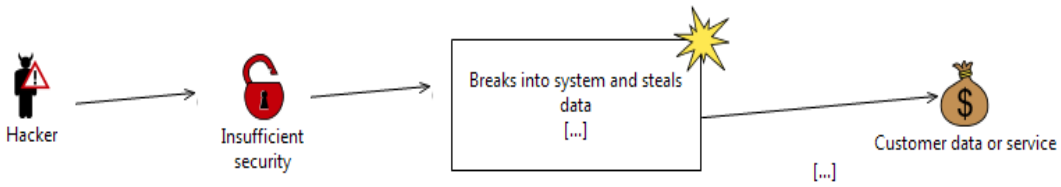

Figure 9: Analysing the Threat Hacking

5.6 RISK EVALUATION

Depending on the priority of the assets and likelihoods of the threats occurring, the threat items can be plotted into an evaluation matrix to document their occurrences. Table 6 depicts this in relation to the threats identified in Table 5.

Table 6 puts forward the consequences of a threat materialising and the impact that it would have consequently (Insignificant, Minor, Moderate, Major and Catastrophic). For instance take the example of T16, this threat has a very high likelihood and if it materialises the impact would be catastrophic. The reason for this is that it impacts confidentiality, integrity and availability of the system. Similarly if we analyse the threat T15, the likelihood of it materialising is high but the impact is insignificant. This threat relates to the distribution of spam therefore the impact on confidentiality, integrity and availability is low.

The likelihood and impact rating is set using the data collected. The impact also denotes the affect the threat will have on the business such as loss of confidentiality can cause loss in trust having the highest impact (Table 7).

		Consequence				
		Insignificant	Minor	Moderate	Major	Catastrophic
Likelihood	Rare	T40	T10	T2,T4,T5,T8, T11, T12		
	Unlikely	T29	T9		T3,T27	
	Possible	T41		T13	T1,T50	T51, T52
	Likely	T15,T34				T16
	Certain	T35				

Table 6: Risk Evaluation Matrix

		Likelihood rating				
Business impact		Very Low	Low	Medium	High	Very High
	Very High					
	High	Confidentiality				
	Medium	Availability				
	Low	Integrity				
	Very Low					

t r a t i n g						

Table 7: Range of Threats for Confidentiality, Availability and Integrity

Threats belonging to confidentiality are classed as high because these have severe effect on trust and the provider's image. Loss of confidentiality can also convert low threats like theft of information to very high. For instance losing unencrypted data is a more severe risk compared to loss of encrypted data.

Loss of availability is relatively classified as medium compared to loss of confidentiality. This is because enterprises are better off using infrastructure provider's resources rather than deploying their own because of the investment involved. Examples include Bitbucket website continuing the use of EC2 even when further attacks are recorded.

Integrity is classed as low because relative to confidentiality and availability the impact is much lower. Loss of integrity can be because of software error, user error, equipment failure and also due to an adversary changing data. From the recorded attacks on Cloud platforms [28], it is difficult to find the reasons for the threats, additionally the VMs can also be restarted and redeployed on different infrastructures to counteract these threats.

5.7 RISK TREATMENT

Once evaluated, the risk mitigation strategies can be generated in terms of the actions taken to resolve them. These can be to accept, treat or outsource the risk. For instance, in a situation of multiple log-ins, the system logs can be scanned to detect this. Once

observed the system administrator can be made aware to take appropriate action on the user account.

5.8 IMPLEMENTATION

Security risk assessment needs to be done at the service deployment and operation stages of the infrastructure provider's (IP) Cloud lifecycle. Figure 10 and 11 describe the architectural details of the risk components involved at deployment and operation stages of the Cloud lifecycle.

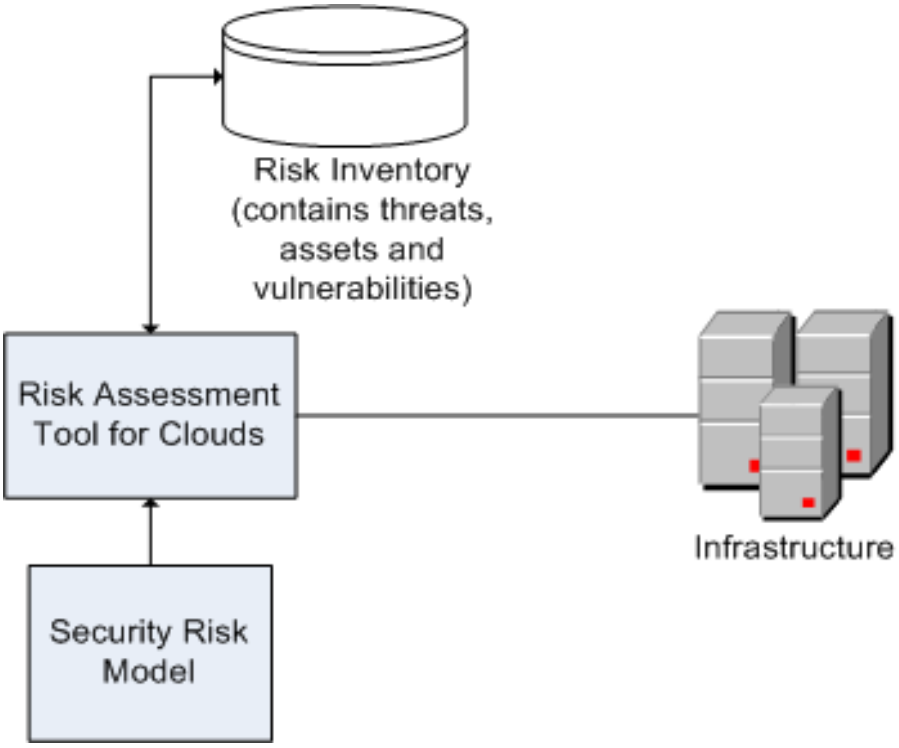

Figure 10: Security Risk Assessment at the Deployment Stage of the Cloud

At the deployment stage, the risk assessment tool will read inputs from the risk inventory which documents all the threats, the vulnerabilities, assets affected and their likelihoods. The risk inventory is based on the threats collected in Table 4. Our risk calculations will use the priority and likelihood values from Table 5.

We now briefly explain the risk calculation algorithm used by the assessment tool. Suppose that both A and B are events; (B|A) is the likelihood of event B given event A has

occurred. For example, suppose that event B represents ‘data leakage’ and event A represents ‘network intrusion’. If we know that ‘network intrusion’ has taken place, (B|A) is the likelihood that data leakage will occur. These dependencies are defined by the Bayesian theorem.

The risk calculation algorithm is given below:

Security_risk_deployment (usecase)

1. Calculate the total number of threats recorded at deployment stage = N

2. For each threat:

    a. Calculate likelihood of event B when event A has occurred = L= (B|A) / 5.0

    b. Calculate relative priority of asset under threat =RP = Asset Priority / 5.0

    c. Calculate likelihood of event B regardless of asset under threat and event A = (B)

    d. Calculate risk index of threat = R = L * RP / (B)

3. Calculate security risk = SR = Sum (R) / N = Sum of risk indices of all threats divided by number of threats found

Based on rules of Bayesian dependencies, the probability of each threat affecting the particular assets can be calculated. For each threat the risk index is calculated by firstly finding out the likelihood of an asset affected 2(a). We already know that the likelihood stays between 1 and 5.

2(b) is relating to the asset priority of the asset under threat. As explained before the priority is dependent on confidentiality, availability and integrity where confidentiality is high, availability is medium and integrity is low. We already have data for priority which is coming from Table 5.

Step 2(c) of the above equation is the likelihood of event B without taking into account the asset. The assets could be customer data or hypervisor code etc. This likelihood is independent to other events for instance from Table 5 we can deduce that the likelihood of "Carrying out a denial of service attack" is 3.

2(d) puts forward the risk associated with this threat. This is done by multiplying the likelihood of event B given event A along with the priority of the asset under threat, to get the impact. We now divide the impact with the likelihood rating of the threat.

To give an example, the likelihood of customer data being compromised could be 4 as per Table 5. Then 'L' = 4/5 = 0.8. Similarly, we then take out priority of the asset which for instance in this case would be RP = 4/5 = 0.8. Multiplying this number with 'L' and dividing it with likelihood of event (B) gives us the risk index of the threat.

Once we have calculated the risk indices of all the threats occurring, we can then sum all of these threats together to come to a number. For instance, we come to the number 5. In case our system finds 4 threats at the time of deployment then security risk for the system would be 5/4 would be 1.25. Risk in this scenario is calculated as a threshold factor rather than a probability. This threshold factor enables us to take decision on taking a mitigation action or not.

As long as the security risk remains more than 1 take no action. The action is taken by admin of the Cloud who is alerted in case a threat is going to be materialised. If it becomes less that one, it means that an alert should be raised, mitigation action needs to be taken place. This is the deployment stage so if the risk is less than 1 then we should address the threats before we move on to the operational stage.

We developed this mechanism using our experience as at times there are threats which are false alarms. In case the number of threats found is too high, the Security Risk would become less than 1 and an alarm would be raised.

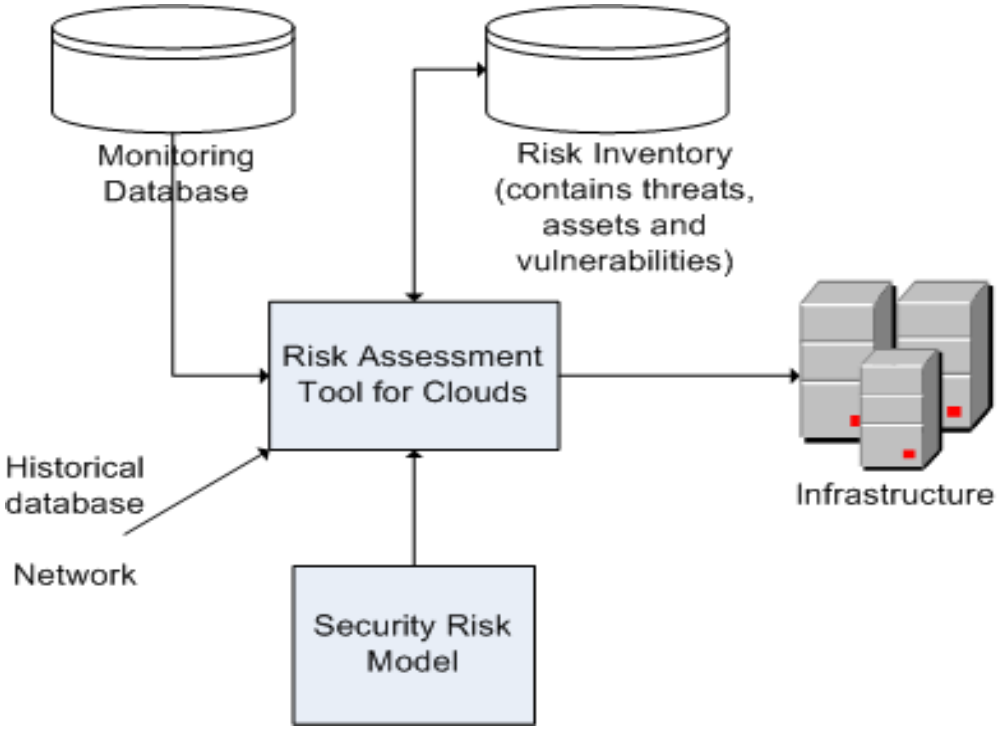

Figure 11: Security Risk Assessment at the Operation Stage of the Cloud

However, at the operation stage, along with the calculated security risk for this stage, the risk assessment tool will be interacting with the monitoring database and additional tools like the network and historical database to monitor if certain threats are becoming live. The historical database can contain details of previously recorded threats that have occurred in the past. The network can include intrusion detection systems and logs which can be parsed to find out if certain events have been recorded [103].

We now continue with the description of the algorithm with the focus on the operations stage. We have taken the value of security risk (SR) from the deployment stage algorithm and will consume it in the operations stage algorithm to deduce relative risk (RR).

Security_risk_operation(usecase) 3. Security risk = SR (Calculated at deployment time) 4. For each threat to be monitored:

4a. Read monitoring inputs

4b. If (event found == true) Event count ++

5. Calculate Event Rate = ER = Event Count / Time Monitored

6. Calculate relative risk = RR = ER / SR

7. If RR <= 1 do nothing, If RR > 1 apply mitigation strategy

In the operational stage presented above, the security risk (Step 3) was calculated during the deployment stage and now we are using is in the operational stage. In step 4, the monitoring inputs like the dynamic analysis being run on operational end points will point to potential threats evolving. Step 4b is a counter which increments every time a potential threat is being found. Step 5 calculates the total event rate, which is the total number of threats found divided by time monitored (time is in minutes). For instance, if 5 threats were found in 2 min this will give us the value 2.5. For relative risk we already have the value of security risk which was 1.25. Now if we calculate relative risk, RR comes out to be 2, therefore a mitigation action is required.

Depending on the value of relative risk (RR), the components can make a decision whether to accept or apply a mitigation strategy stored in the risk inventory to compensate for the risk. The risk is mitigated during the same time period.

Figure 12 shows the output of 20 simulated samples collected while executing the risk model during the operation phase. Depending on the event rate per sample the relative risk can be calculated according to the algorithm step 6. If the relative risk is less than 1, the software can choose to accept the risk but if higher, the mitigation strategy will get activated which may ask for human intervention as the risk is going high. It is pertinent to mention here that relative risk is not calculated as a probability but rather a threshold. It is a relation that exists between security risk calculated at the deployment stage and event rate found at the operational stage.

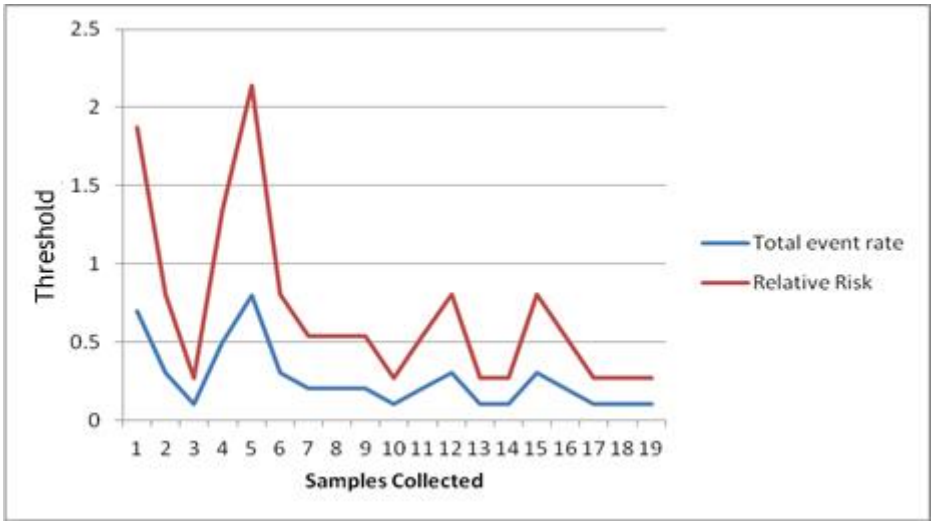

Figure 12: Calculating Relative Risk using Samples and Event Rates. An Action is taken when Relative Risk is more than 1

The figure above shows the relative risk where the value goes over 2. As we have explained before the risk is not calculated in the traditional manner. We calculate RR to understand the co-relation between potential threats occurring compare with a specific threat. For this we set the threshold of 1 whenever the value 1 is breached there is requirement to take action. These calculations should not be taken in the traditional sense that risk cannot be more than 1. The data for the above calculation originates from a simulator which was run for the OPTIMIS project. The simulator calculated the values using the formulas explained above.

The simulator was implemented using Java code, it was part of the 'Security Risk Framework' for OPTIMIS project. The simulator has four main parts, the first one is the monitoring database. The monitoring database is used to hold information relating to potential risk coming up in the assets relating to the OPTIMIS toolkit. These assets are networks, computers, servers etc.

The second part of the simulator is the risk inventory where all the vulnerabilities, threats and assets are registered. This part of the simulator provides the risk framework the ability to find out which are threats are related to which assets.

The third part is the historical database which provides information about threats and their severity from a historical perspective. For instance, the threat of 'data leakage' is high and the likelihood is high as well. This information will be used to calculate the impact.

The fourth part of the framework is the code which runs a Linux server. It orchestrates the whole process that has been explained before. The details of the code refer to the index for code of the 'Security Risk Framework'.

5.9 CONCLUSION

From the threat analysis performed, we have shown that the information security principles of integrity, confidentiality and availability are most relevant to the Cloud related scenarios. The information risk ratings performed shows the loss of confidentiality is rated as the highest level of risk followed by availability and integrity.

The risk model presented here allows monitoring threats based on the events being logged by the detectors leading to a calculation of the relative risk. However, a fine grained analysis needs to be performed on threats which are difficult to detect via certain events or have a cause and effect relationship to other threats. These may be more specific to confidentiality or integrity classifications of the threats. Further future work includes testing this system on a Cloud platform with monitoring agents installed which will log certain threats when they occur. This will then be extended to work on determine threats which may be eventually seen based on the data being collected and difficult to determine directly from the events. Finally, the results from the testing and evaluation, advocate that the risk model does correctly assess and prioritize the risk.

CHAPTER 6

SECURING SCALABLE VIDEO IN THE CLOUD

6.1 INTRODUCTION

Scalable video using Cloud computing is a potential solution for the distribution of media content to a large number of users. This may occur over a heterogeneous network connected to devices with different capabilities and diverse set of users. Although some of the problems are well known and understood in information and network security, there is still a need to improve the existing solutions to produce a solution that is both adequately secure and efficient in highly distributed and scalable environments. In this Chapter, we describe such improvements using a Cloud computing scenario where video content is made available through a Cloud platform.

The author has published a patent [157] which explains in detail the authentication and encryption of scalable video in the past. The patent came out of MSc thesis submitted by

the author at the University College London. This work is an extension of the prior research work.

When put on an Infrastructure as a Service (IaaS) Cloud video content should then be viewable by different consumers using different levels of bandwidth and security requirements depending on their identity. This requires a mechanism through which a Cloud service could be authenticated and encrypted by end users. This Chapter describes the novel solution of securing scalable video in the Cloud discussing the various threats for video distribution and how these can be made more secure in terms of confidentiality, availability and integrity, particularly through source authentication and encryption.

Scalable video is a form of distributing media, as Cloud computing paradigm is built around large number of users accessing a centralized service, the scalable video scenario fits very well within the Cloud computing paradigm. In this case, where the broadcast medium is a video, it is imperative that the data owner ensures the authenticity, integrity and the confidentiality of the broadcasted video. This requirement does not only conform to the basic compliance requirements but is also necessary to get any economic return on the video that is broadcasted to prevent any copyright violations.

Presently, it is increasingly popular to broadcast media such as pay-per-view or subscription video over the Internet but this lead to various security problems. One main practice to secure the video is broadcasting it as encrypted files. This allows only the paying subscribers to access the video, and the non-paying parties not being able to view the content. Conventional encryption techniques, such as those used for satellite broadcast TV, are often based on the premise that each subscriber is likely to be a long term subscriber and may invest in new hardware such as a set top box to subscribe to encrypted channels. In such circumstances it may simply be possible to provide a shared encryption key to each subscriber, changing the key at regular intervals.

However, there are new problems arising with models of Internet video distribution where subscribers may wish to subscribe quickly to watch a single video and then unsubscribe immediately thereafter. For considering subscription services, it is desirable

for the provider of the encrypted video system to prevent any former subscribers from being able to decrypt the video with any past distributed keys provided during their subscription periods. It is also imperative that non-subscribers do not have access to the broadcasted video.

Key chains can be used as security tokens for accessing data. Various methods can be used to process key chains such as TESLA[158] that can interlink keys in a manner where the last element authenticates the first element of the next key chain. Digital signatures are only used at the start of the first key chain, with all subsequent chains interlinked in such a way that using digital signatures at the start of the each key chain is not required.

This Chapter discusses the various threats involved for video distribution from the perspective of using Clouds to distribute scalable video. It focuses on the three aspects of security- confidentiality, availability and integrity – showing how these can be influenced in these scenarios discussing each in detail.

This research work in this chapter is focused around the research gaps of RG1 and RG4. The RG1 research gap is to do with providing scalability and confidentiality in Cloud computing scenarios. Also this work links up with the research gap RG4 which identifies the problem of user revocation in Cloud computing scenarios.

6.2 BACKGROUND AND RELATED WORK

To cater for the needs and capabilities of end users over various heterogeneous networks, scalable video is a potential solution for the distribution of media content. Scalable Video packets are divided into two parts. The first part is called the base and contains video information that is of low resolution and quality. The second part is called the trail. This part has video packets which are used to enhance the resolution and quality of the video.

Broadcasting of video is carried out by a video server with scalability servers appearing between the video server and the end-users. The job of the scalability servers is to truncate the trial part of the video packet with respect to the bandwidth requirements

and the capabilities of end-users. Trial truncation decreases the quality of video received by end-users.

Previous work for scalable video has looked at encryption, key distribution protocols and authentication in video scenarios. Apostolopoulos et al. [159] have discussed a mechanism which encrypts scalable video packets. In [160], the focus of their work was to find the most efficient symmetric encryption algorithm with which video packets could be encrypted. The authors analyzed AES (Advanced Encryption Standard) and Triple DES (Data Encryption Standard) showing that for Triple DES, the overhead is between 2-2.5% and for AES it is 7%. [161] presents an idea of progressive encryption also using scalable video. Even if intermediary servers truncate scalable video packets, decryption will still be possible. The BIBA signature scheme [157] works by using self-authenticating values that are linked with a public key. So given a public key, it is not possible for an adversary to compute the self-authenticating values also known as seals.

However, in all of the prior work no explanation is given on how the keys will be distributed in a scalable video scenario for encryption. Additionally various assumptions are made such as assuming the sender and receiver will already have the keys for encryption. Blakely[162] has discussed the concept of secret sharing among various users. An example explains the m-out-of-n threshold secret sharing scheme which allows for a secret message M to be distributed as a selection of n shares {s1,…,sn}. This allows two properties to be achieved (i) any collection of m shares is able to reconstruct the secret message M; and (ii) Any subset of (m - 1) or less shares reveal no information about M.

6.3 TESLA

TESLA (Timed Efficient Stream Loss-tolerant Authentication) is the protocol used for authentication having low communication and computational overhead, scaling to large number of receivers. Its main advantage is the use of key chains. TESLA [158] uses the

initial part of the authentication key by securely sending it and then subsequently the rest of the keys are authenticated.

The main idea of TESLA is that one uses a symmetric key to compute a MAC value at a time when they alone only know the key. In the next time slot the key is made public to all other parties who carry out the authentication. In this way, parties carrying out the authentication know for sure the true identity of the user which computed the MAC. Without this time delay, everybody would have the symmetric key and could have computed the MAC.

In TESLA, the users who receive a message at time i and the key at time i+1, need to be able to verify that the key they received at time i+1 is a valid key that belongs to the user that has sent it. For this they use key chains. The sender starts by generating a random key km. From this a chain of keys is computed by applying a hash function h.

The end of the chain (k0) is distributed to all the receivers in a secure manner, such as by using digital signatures. The security of the protocol depends upon the last part of the key chain which is k0. At time t=1, the sender sends k1 (which was used to compute a MAC in time slot 0) to all the receivers, they verify that it is the correct key by verifying that h(k1) = k0.

In this case, they are assured that k1 is indeed the correct key. This continues for all the keys, until the root (km) is used, and a new key has to be generated and distributed.

Using one way chains is advantageous for authenticating a packet without much overhead in terms of computation and bandwidth usage.

6.4 SCALABLE VIDEO

As seen from [163] scalable video enables the recipient of the video to reconstruct lower quality video from lower bits. The packet of the scalable video has got two parts, base and trail. The more the trail part is truncated the lower the quality of the video is. If only

base part is received then the quality of the video is lowest. This enables recipient of the video with different network limits to receive the same video but of different quality.

6.5 SECURITY ISSUES ON THE CLOUD IN GENERAL

To overcome the hurdles of security, the UK government has proposed to promote the use of open source software as part of its G-Cloud program for delivering ICT to emerging suppliers [164]. The National Institute of Standards and Technology (NIST) group has proposed a list of security risk and mitigation for a lifecycle to be followed for performing risk assessment[165] discussing the certification and accreditation for threats in accordance with the government laws analysed per stage with a detailed analysis [166].

For a scenario of subscription services, it is important that the broadcast video will only be transmitted in an encrypted manner. This work only considers the base component of scalable video packets as being encrypted. This allows for a simpler system with less information being encrypted whilst still providing the system with the security requirements. This is because without the decrypted form of the base layers no video could be obtained from the trail layers alone.

6.6 THREATS AND ASSETS THAT NEED TO BE PROTECTED

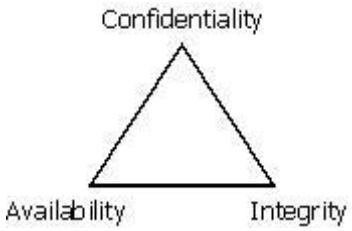

Figure 13: Security Triangle

Security can be broken into three main aspects (figure 13) which cover it, namely – availability, which means data is available when needed, integrity, which means the data is not modified without being detected and confidentiality for the disclosure of data to unauthorised parties. Unlike normal grid computing, using Clouds presents additional

threats to be considered for security reasons. For instance, data access is not a huge threat on grids, but in Clouds, because the data is hosted geographically at different locations, this is an important factor. This is particularly relevant in terms of video broadcasting as videos can be delivered as similar files on the Cloud. Therefore the main hub holding the video data needs to consider the geographical location and the access rights to it, for safety of the data. Various authentication models can be introduced to make each threat more secure as a mechanism to overcome them. Khan et al. [138] have discussed the six main categories of threats that can summarise all the kinds of threats that Cloud Computing faces summarized in Table 8 [167].

Clouds involve a three stage process namely – pre-deployment, deployment and operation.

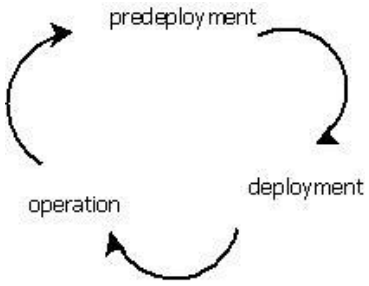

Figure 14: Service Lifecycle for Scalable Video

Distributing video over the Cloud involves a number of stages (figure 14). During the pre-deployment stage, the video is set up on the video server and encryption keys are generated with user subscription being set up. During deployment, the keys are encrypted and sent across.

Threat Category	**Specific to Cloud scenario**
External attacks	Threats to Confidentiality, Integrity and Availability. Includes all kinds of threats for public use. Examples include hacking attacks.

Threat Category	Specific to Cloud scenario
Theft	Threats to Confidentiality, Integrity and availability. Cloud computing supports multi-tenancy using same resources which causes threat to data hosted on the infrastructure.
System malfunction	Threats to integrity. Some software used extensively on Clouds has bugs or malfunctions.
Service interruption	Threats to availability, integrity and confidentiality. Unavailability of service or data due to DoS attacks. Natural disasters can cause this as well.
Human error	Threats to confidentiality. No control on how vendors use the system. This is difficult to control and may not directly apply to video distributing.
System specific	Specific to user. May not apply directly to video broadcasting.

Table 8: Threats Categories

Table 11 discusses in detail a full list of threats that should be monitored in terms of video distribution. The values for likelihood and priority of the assets were taken from the IRAM tool which has been used previously for Cloud computing scenarios. Learning from the past experience of undertaking risk assessment of Cloud scenarios priorities were set. Depending on the priority of the assets, the likelihood (Table 9) and the priority of the threats can be assessed. This can produce a likelihood and impact rating. The impact

shows how this threat may also affect the business of video distributions (Table 10). More information on this analysis is provided in section 6.10.

		Consequence				
		Insignificant	Minor	Moderate	Major	Catastrophic
Likelihood	Rare		V5, V12, V9, V11	V2,V4, V10, V6	V18	
	Unlikely	V15, V19	V7		V14, V21, V22	
	Possible	V17		V20	V8, V13,V23	
	Likely	V16				V1
	Certain					

Table 9: Risk Evaluation Matrix

	Likelihood rating

Business impact rating		Very Low	Low	Medium	High	Very High
	Very High					
	High	Availability				
	Medium	Confidentiality				
	Low	Integrity				
	Very Low					

Table 10: Range of Threats for Confidentiality, Availability and Integrity

Some of the threats which do not apply to the video broadcasting scenario, from the general Cloud Computing scenarios are as follows:

- Isolation of tenant application. Affects integrity, confidentiality and does not apply to video broadcasting.
- Data encryptions. Applies to all three availability, confidentiality and integrity and is already covered in the key authentication process during the pre-deployment process.
- Data segregation. Affects the availability and integrity also does not affect broadcasting issues.
- Tracking and reporting service effectiveness can be given by customer review and end-user experience affecting the credibility of the server.
- Compliance with laws and regulations of copyright issues, contract breach. Affects the confidentiality and integrity of the business during the pre-deployment stage.

In the scalable video scenario, threats belonging to availability are high priorities that need to be protected, because they affect the business integrity of the video servers. DOS attacks to the hypervisor are the most common threat. The next highest threats would be the confidentiality for the user data paying for the service. Integrity of the service is classed below the above two which relate more to the business impact of the video server because of the software errors and equipment failure.

6.7 AUTHENTICATION METHODOLOGY

The solution is divided into two parts source authentication and source encryption. In source authentication, the video is authenticated by the subscribers. The source encryption, the video is decrypted. Furthermore, the process of key management will be explained for the scenario.

Figure 15 shows the system architecture with all the entities and their communication interactions.

The system is composed of four different entities – the Video Server, Scalability Servers, Subscribers and a Key Distribution Centre (KDC).

The Video Server is hosted on the IaaS Cloud where the source of the video gets encrypted and then broadcasted. A MAC is also appended to encrypted video packets, so that they can be authenticated by Subscribers. TESLA will be used to provide authentication in the system. The video is then broadcast to the Scalability Servers. The Video Server also generates keys that will be used for video encryption. These keys will be given to the KDC through a secure channel such as TLS or IPSec.

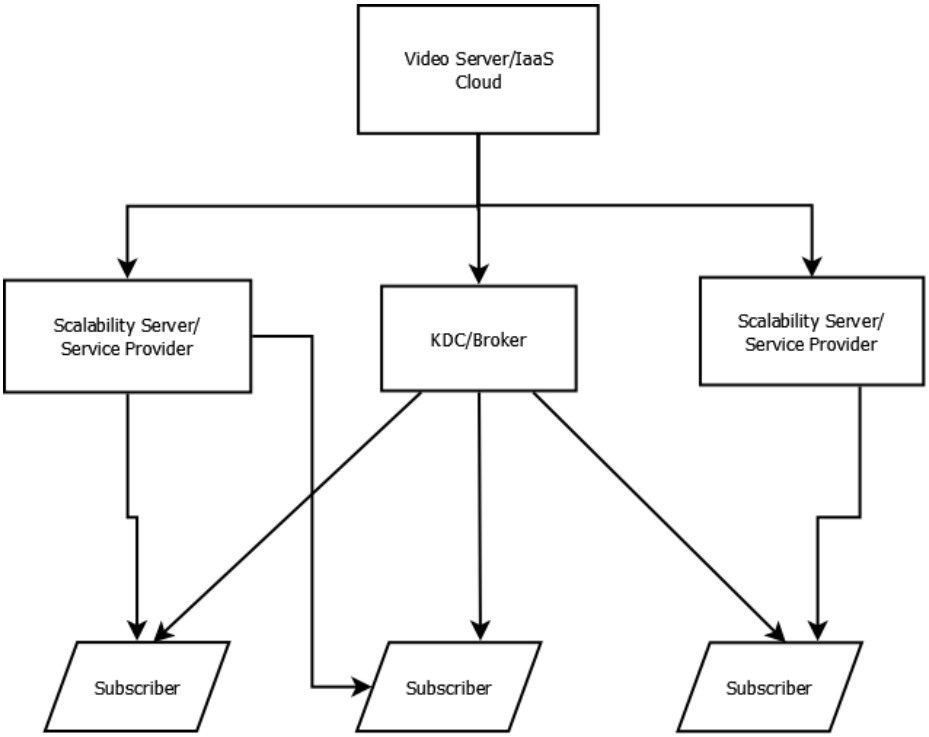

Figure 15: System Architecture Application Scenario

Scalability Servers role can be taken up the Service Provider from the Cloud computing perspective. The Service Provider would use IaaS Cloud infrastructure to host the video. The Scalability Server hosted on the Service Provider would receive video from the Video Server on the IaaS Cloud. Scalability servers truncate video packets relative to the bandwidth requirement of Subscribers to which they will forward the video.

The Key Distribution Centre plays a pivotal role in the whole system. The KDC can be hosted on the Broker, where the Broker ensures that it distributes keys related to authentication and decryption keys to all subscribers. It interacts with the Video Server to get the keys and also authenticates new incoming subscribers to the network.

The Subscribers are the end users of the system who receive the video. They also authenticate the source of the video using the extended TESLA authentication protocol.

6.8 SOURCE AUTHENTICATION AND ENCRYPTION

Figure 15 describes the process of authentication of broadcast video being carried out. Authentication is needed so that end users can verify the identity of the video source[157].

Different steps are required in the overall authentication process carried out by the video servers. These include the generation of a first authentication key, generation and storage of a key chain using one or more one way functions and use of these keys for the authentication of broadcast video data. Key chains are created by following the TESLA protocol using suitable time intervals and appropriate one way hash functions.

6.8.1 AUTHENTICATION OF VIDEO PACKETS

The process of authenticating video packets originating from a video server is the following.

A first key Ki is generated by the video server. This may be generated randomly and can be of a predetermined length. Next, a one way hash function F is used to generate the next key Ki-1 where Ki-1=F(Ki). As F is a one way hash function, it is very difficult to determine Ki from Ki-1 but it is easy to generate Ki-1 from Ki provided F is known.

The remainder of the key chain is then generated by further applying the one way hash function F until a final key K0 is generated. The function F is applied i times to generate K0, so K0=F(K1)=Fi(Ki). Each key can thus be generated from previous keys through application of function F an appropriate number of times.

Consequently, all the keys in a key chain K can be generated from Ki, but Ki cannot be calculated from any other key. This is because one-way functions are used. As a result Ki will remain secret and unshared until it is revealed.

Using the properties of key chains it is easy to verify that different keys belong to the same key chain e.g. checking that a key Ki -x is indeed the xth key in a key chain can be done by checking that Fx(Kx)= K0.

6.8.2 INITIAL AUTHENTICATION SETUP

All the keys in the key chain K, their order and the hash function F are stored in memory on the Video Server. The final key K0 is then sent to the Key Distribution Centre using a secure path which can be achieved using TLS, IPSec. The video server also determines a time scale for using key chain K and determines the time intervals with which it will move from using the final key K0 to each further key along the chain.

The KDC then calculates the maximum time intervals at which recipients should receive a video packet from the video source with a given key based on the time periods calculated by the video source and adding the expected time delay to reach the recipient. The recipient can be scalability servers and/or subscribers depending on the implementation.

Where both scalability servers and subscribers authenticate, different maximum times will typically be calculated for each based on any assumed or determined knowledge of communication paths. The maximum times for different subscribers can also vary depending on network paths with which subscribers are connected.

The video server also generates a message authentication code (MAC) using symmetric encryption and final key K0 to authenticate video packets. This MAC is attached to a base video packet and transmitted to scalability servers. A counter is also started for the number of keys sent and is set at “n=1”. The final key K0 is sent by or requested from the KDC using a secure asymmetric system such as IPSec. The recipient of video packets can then check that the KDC and the video server correspond by decrypting the MAC using the final key K0 and using the secure asymmetric protocol or authentic channel prevents a third party from transmitting this information. The hash function F is also provided to the recipient and/or is stored at KDC and requested when needed. The hash function F can be provided to the subscriber by saving it or hard coding it into the subscriber end devise rather than transmitting it remotely. This option reduces the likelihood of a third party gaining access to the hash function.

6.8.3 SUBSEQUENT AUTHENTICATION STEPS

After a period of time, the video server generates a second MAC using the next key along the key chain in the "revealing" order which on first application is key K1. This is then attached to a scalable video packet and transmitted to recipients. After the predetermined length of time which may be time delay d, no video packets from the video source will use the first MAC calculated from key K0. The latest key used for creating a MAC, in this case K1, is also sent to the recipients by the video server or KDC.

The recipient then decrypts the MAC using the key - in this case K1, and checks that the message is correct. The recipient then applies the hash function F to the latest key K0, the appropriate number of times (which in the first case is once) and checks that this results in the final key K0. The recipient also checks that the time at which the video packet was received was done so within a time from receiving the base packet that is less than or equal to the maximum time allowed for that particular recipient calculated and stored by the KDC.

On subsequent times the importance of checking the time delay becomes clear. Once a key Kn is revealed to recipients, any third party posing as a recipient, that knows the one way hash function can generate all keys below Kn on chain K. In practice a recipient may temporary loose contact with the video server by intention or through missed data packets. A third party which may have received these packets can then use the keys to fool the recipient that packets from the third party using these keys are authentic. This is prevented by the time delay check since after a period of time the video source will no longer use a key Kn and the KDC will inform the recipient that a given key Kn was received outside the maximum time delay and is therefore not to be trusted. Accordingly this TESLA chain uses time to produce asymmetry and therefore security, even though the keys and encryptions are symmetric. It also has the advantage that when intermediate values in the chain are not received, all keys can be authenticated from the final key K0 and the correct number of applications of function F.

6.9 CASE STUDY: GENERATING KEYS FOR USER ACCESS

In this section we describe a method for generating and updating keys for the encryption of video broadcast from a video source in our scalable video scenario. Our method uses secret sharing as this prevents a single user or a group of users to know the value of encryption keys. This allows the system to be more secure, preventing non-users or former users from learning new values of encryption keys when these are updated.

The process of updating keys will use the method of secret sharing [136]. This will allow for the encryption key to be split into a number of shares each of which will be distributed over the different subscribers. This is done to ensure the secrecy of the encryption key. In the case, the encryption key of the video broadcast needs to be changed, the KDC will communicate with each of the subscriber devices about this. The corresponding information will also be sent to each of the devices. The new encryption key can then be reconstructed by groups of participants interacting between them.

6.9.1 INITIAL SETUP

Upon initial setup, the KDC will carry out a grouping of all the subscribers. For each of the new subscribers the KDC stores details of the group to which they belong along with a hash function specific to the hardware of the subscriber service hardware. The KDC will also store the share and hash function each subscriber hardware stores in memory.

The KDC will then carry out a secret sharing of the encryption key which will be used for the encryption of the video to be broadcast. For each of the subscriber groups a different secret sharing of the encryption key will be carried out using an n-out-of-n secret sharing scheme – with n denoting the number of subscribers in a subscriber group.

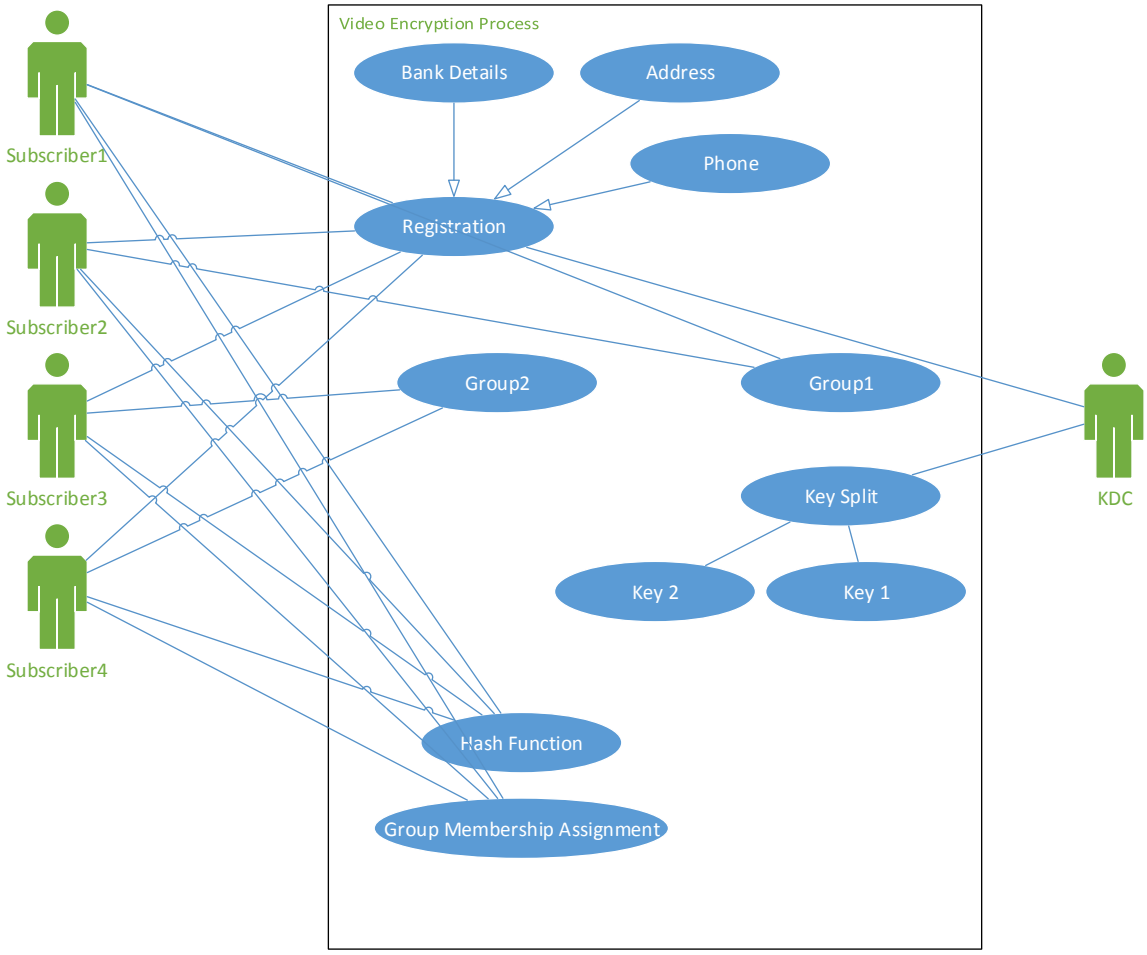

Figure 16: Use case diagram for the video encryption using secret sharing

The Figure 16 explains that various process using the UML format that takes place relating to subscriber registration to distribution of hash functions and group membership assignment.

The KDC proceeds to send the following to the broadcast server and each of the subscriber hardware in an authenticated manner using digital signatures.

The KDC informs the broadcast server in an authenticated and encrypted manner the value of the encryption key to be used for encrypting the video to be broadcast.

In parallel, for each subscriber group the KDC sends a single share of the encryption key to each of the subscriber hardware (from the corresponding secret sharing of the

encryption key) sending each share only once. The KDC also informs each of the subscriber hardware the other subscribers which belong in the same subscriber group.

After this, the subscribers can interact between them (outlined in the next section) in their appropriate subscriber groups in order to reconstruct the encryption key which will allow for the encrypted video to be decrypted (figure 17).

The minimum number of subscriber associated with each group should not be less than 10. We came up with this number because it will be hard (financial perspective) for an adversary to find out the keys of each of the subscriber. An adversary would have to spend substantial amount of resources in finding the key parts which is not financially feasible as it would be better for him to just pay for the service. Moreover, the encryption key will be updated every 24 hours even if no subscriber leaves the system. Therefore, it would be useless for an adversary to find out the decryption key as it would be useless in 24 hours when it would be updated.

Figure 17 demonstrate the process of user registration and assigning of key shares to each subscriber.

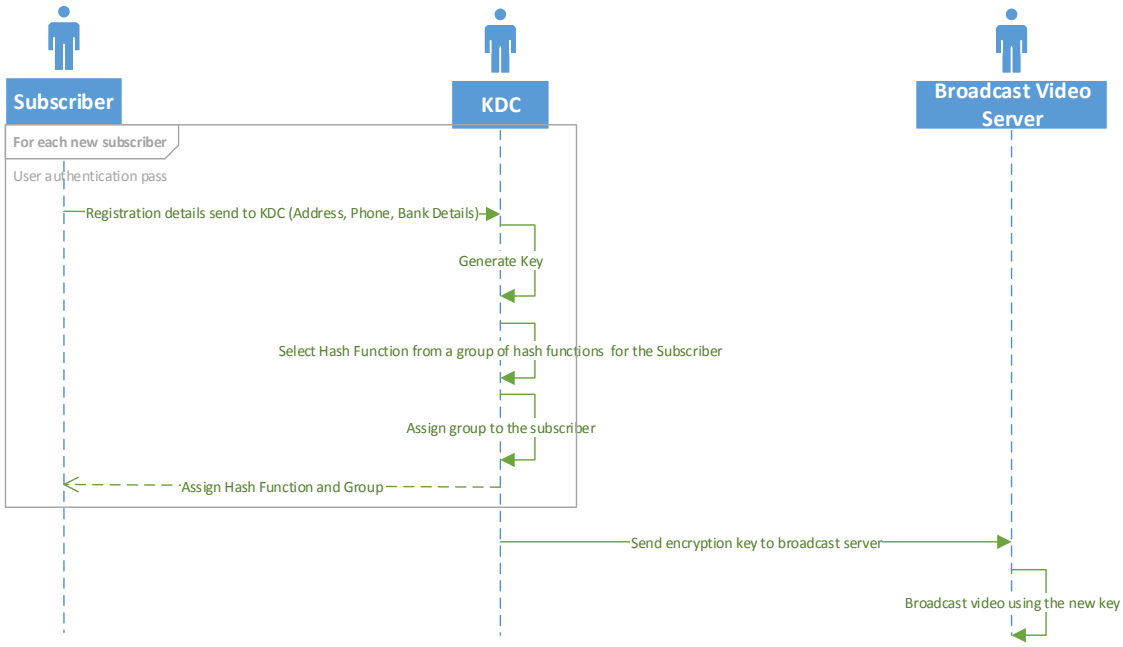

Figure 17: Sequence diagram for the subscriber registration

6.9.2 UPDATING ENCRYPTION KEYS

Updating of encryption keys occurs when a subscriber leaves the service or when a new subscriber enters the service. The encryption key may also change whenever the KDC deems that an alteration of the encryption key is necessary, for example when a period of time has passed without the encryption key being altered. As explained before, we recommend that key should be updated every 24 hour at least.

When a key needs to be updated, the KDC sends a digitally signed signal to each of the subscribers that a key update will be carried out. Each of the subscriber hardware then proceeds to carry out a hash of the share (of the previous key) stored in memory. As the KDC knows the hash function and share stored on each of the subscriber hardware, the KDC can do the same and thus be aware of the result of this hash operation.

For group sizes where the total number of subscriber are not large (50 or above), it would be required to generate hash functions that can be associated with subscribers. We understand that dozens of hash functions are readily available like SHA0, SHA1, SHA2, MD5, Skein, Keccak, Radiogatun and their extended flavours. For cases where the requirement is to have 50 or more hash functions, it would be required to develop a function that can produce hash functions. Moreover, for this scheme where scalable video is broadcasted, we envision that the group sizes would be limited to geography therefore group sizes would be between 10 to 20 subscribers each. Therefore the requirement is not there for our current scheme to develop a function that generates hash functions.

The KDC will proceed to select a new encryption key and carry out a n-out-of-n secret sharing of the key for each of the subscriber groups (where n denotes the number of subscribers in a subscriber group). The KDC also sends in a secure manner the new encryption key to the video broadcast server. Figure 18 presents the system setup.

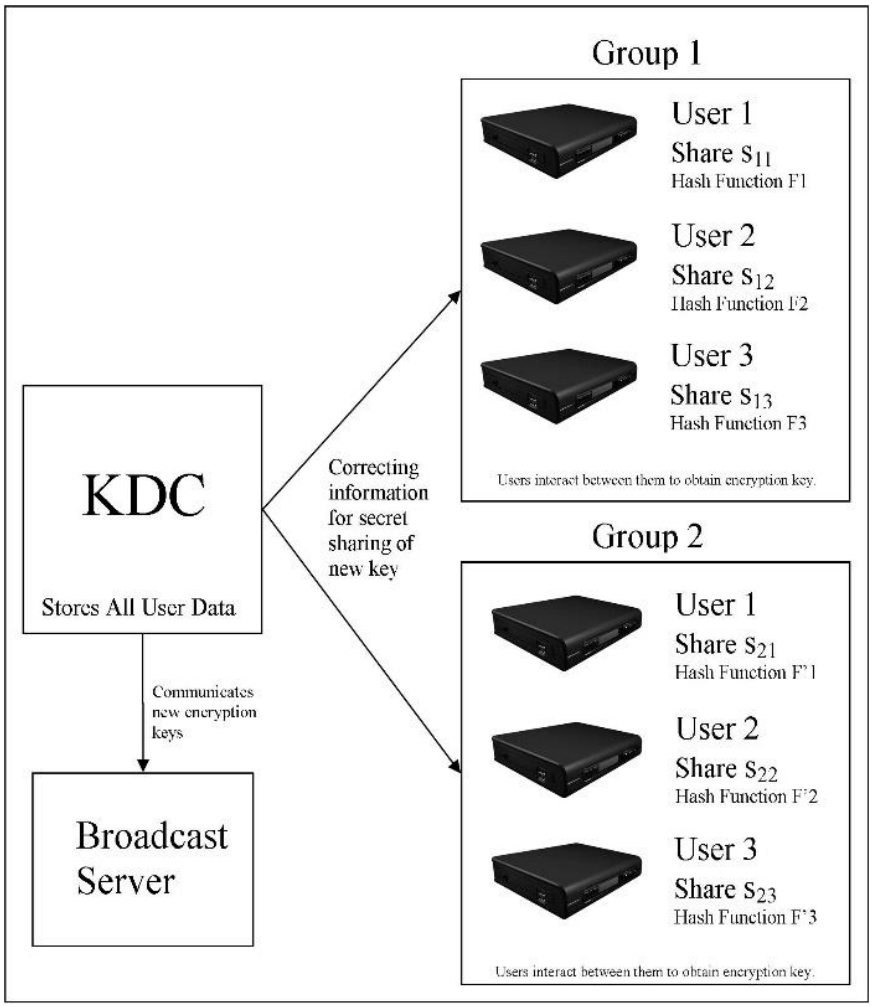

Figure 18: System Setup

For each subscriber the KDC calculates the difference between a share si of the new key (using each share only once from the corresponding subscriber groups and secret sharing) and the result of the hash operation rh corresponding to the subscriber. The KDC thus calculates di = si - rh which is authentically sent to the corresponding subscriber. The KDC also stores the new share which corresponds to each of the subscriber hardware. Upon receiving this value, the subscriber hardware calculates the new share value (si = di + rh over a finite field).

If there have been any changes to the original subscriber groups, the KDC also informs the appropriate groups of the alterations. If the key update occurs because a new

subscriber enters the system, the new subscriber hardware is primarily sent a random value – a hash function will originally be present in the subscriber hardware.

After the hardware of all subscribers have calculate the new share values, the subscriber groups can interact between them so that the new encryption key can be calculated. The figure 19 demonstrate the process that is used when a user revocation takes place.

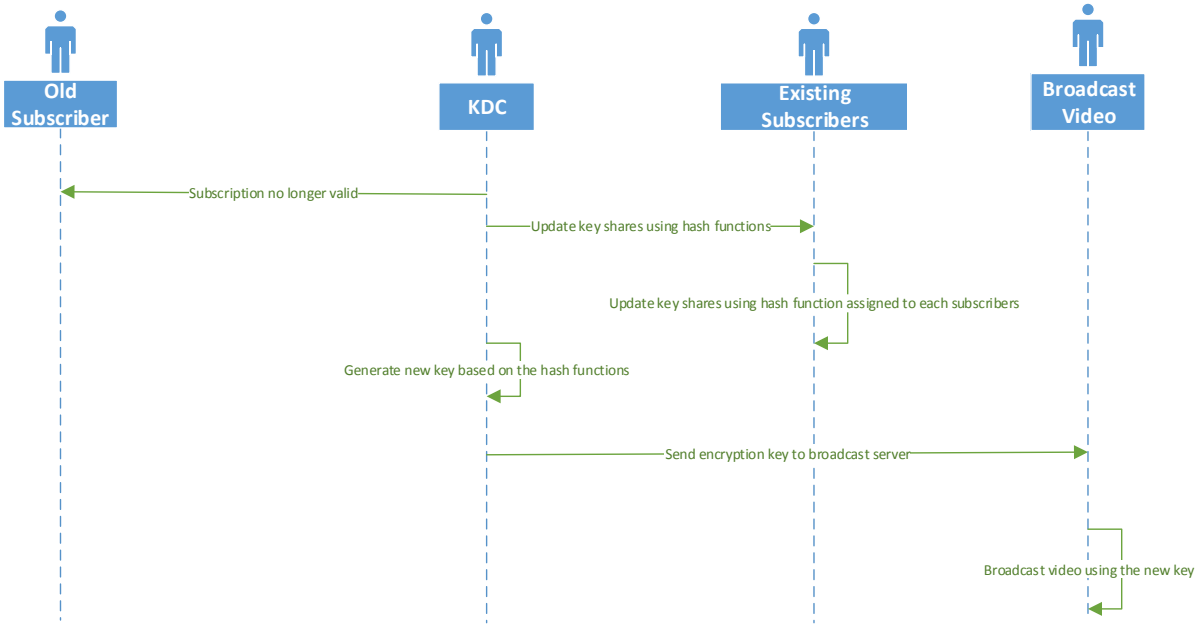

Figure 19: Sequence diagram for user revocation and new key generation

6.9.3 GROUPING OF SUBSCRIBERS

A group of subscribers can be anything appropriate to the application setting. A group of subscribers could be all the subscribers within a town or neighbourhood for example. As the secret sharing of the key is done using an n-out-of-n protocol. This means that the shares of all subscribers in a group need to be pooled together to reconstruct the encryption key. We thus assume that the set top boxes of all subscribers are always available to carry out the appropriate actions for the reconstruction of any new encryption keys.

As explained before the total number of subscriber for this setup should be no less than ten. However, the size of the group is dependent on other issues such as the total number of subscribers and total number of subscriber in that geographic location etc. A group with only one subscriber cannot exist as this would fail at the registration phase. Moreover, groupings are done by the KDC, therefore it has control over who gets in which group.

6.9.4 SECURITY OF VIDEO ENCRYPTION

The method of updating encryption keys is secure against any non-paying subscribers to view content for free. Security here refers to a level of security which to be broken requires non-paying subscribers to carry out enough effort (to learn encryption keys) equivalent in value to the subscription fee of the service. In this sense, the cost of learning the secret encryption key and paying the subscription fee are equivalent.

There are two different types of non-paying subscribers – the non-subscribers of the service and the former subscribers of the service, and prove the security of the system for each.

For the security of the system against non-subscribers, the non-subscribers were never part of the service and are not aware of any of the encryption keys or share values. The non-subscribers can listen to the communication of subscribers. Due to the key update scheme that is used, they only thing that will be learned are corrections sent by the KDC upon old shares held by each subscriber. As the secret sharing used is a n-out-of-n scheme, this means that such an attacker to the system would have to listen in on the traffic of all the subscribers of a particular group.

Additionally, even if the attacker was able to listen to the traffic of all the subscribers in the group this would not permit them to learn the encryption key. This is because the attacker is not aware of any of the original share values that are present in the memory of subscriber's set top boxes. Because of this, no matter what information the attacker may listen to and as the secret sharing of the key is carried out using a n-out-of-n secret

sharing scheme, this stills allows for all the encryption keys to be possible. The encryption key is thus kept secret against such kind of attackers.

For the security of the system against former subscribers of the service, the security is guaranteed because of the n-out-of-n secret sharing scheme of the encryption key. Using t as the number of former subscribers, that are attacking the system and assuming in the worst case that they are working together, the encryption keys when all these former subscribers were part of the service were secret shared using a (n+t)-out-of-(n+t). Because of this and assuming that these t former subscribers learned the value of the encryption key, this means that the set of t attackers know t+1 points (or shares) of the polynomials used in the secret sharing of an encryption key. Using this information, the t attackers (assuming they have infinite computing power) can find the q(n-1) possible polynomials that could have been used in the secret sharing of the encryption key – where q denotes the size of the finite field used in the secret sharing schemes.

For each of these polynomials, the attackers can learn the shares of all the non-attacking subscribers. Assuming that the value of q(n-1) is very large and even if the attackers of the system were able to listen to all the incoming traffic of the n paying subscribers, this would still leave all the possible encryption keys as potential keys when a key update occurs. As a result of this, the attackers do not learn the encryption key used and thus the system is secure.

6.9.5 MATHEMATICAL FORMALISATION

In this section an example is provided of how mathematically a key will be calculated using the protocol explained above. The example is for three key shares between three subscribers.

In order to keep the example simple the KDC will make a group key and divide it into three parts. The generalized equation for the secret sharing is as follows:

$D(x)=a_0+a_1x+ a_2x^2+\ldots+a_{k-1}\, x^{k-1}+ a_kx^k$---------(1)

The KDC will make an equation by which it will calculate the shares, for example the equation can be:

$D(x)=a_0+a_1x+ a_2x^2$ ----------(2)

The equation will be set by the video server initially. But afterwards it will be calculated on the basis of the shares values that come out after applying the hash functions:
Now using equation 2:

$D(x)=1+2x+ 3x^2$ ----------(3)
The above equation is of the second order and it will be made by the KDC. We are taking 3 here as an example only.

Now the KDC will calculate the three shares which will be:

D(0)=1

$D(1)=1+2(1)+ 3(1)^2 = 6$
$D(2)=1+2(2)+ 3(2)^2 = 17$
$D(3)=1+2(3)+ 3(3)^2 = 34$

Now the secret shares will be:

(1, 6) plus Hash function F1

(2, 17) plus Hash function F2

(3, 34) plus Hash function F3

Whereas D(0)=1 is the group key and if we put all the three shares together, only then we will be able to get the group key. The three shares will then be sent to the KDC along with the group key. Now the KDC will assign each subscriber with one share of the secret. In this case there will be only three subscribers. In order for them to get the secret key out they will have to collude together. So two of the subscribers will send their secret to the third subscriber who will now know of all the secrets. What it will not know are the

hash functions of the other two subscribers which will stop it from generating future keys. The following equations will be used by the third subscriber to get the key out,

$a_0+a_1+ a_2 = 6$ ----------(3)

$a_0+2a_1+ 4a_2 = 17$ ----------(4)

$a_0+3a_1+ 9a_2 = 34$ ----------(5)

In order to get the group key out the subscriber S3 will 3,4,5 simultaneously

$a_0 = 6 - a_1+ a_2$ ----------(6)

substituting 6 in 4 ad 5

we get $a_2= 3$ and $a_1= 2$

we put the above values back in 6 we get the group key which is

$a_0 = 6 – 2 – 3= 1$

So 1 is the group key.

The third subscriber S3 will then send the group key to all others in the network which in this scenario are S1 and S2.

Now let say that the subscriber S3 leaves the network and a new subscriber comes in the network S3` taking its place. But since S3 knew the group key, so it is now needed to update the group key. The S3` will get all the information from the KDC. It will be given the secret shares which S3 used to have-that were (3, 34) and hash function F3. The KDC will send a signal to all the subscribers that the group key needs to be changed so each subscriber using its own share of the key and hash function will generate a new share secret or part of the key. All the new shares will then be sent to S1, so that it would be able to compute the group key.

We will choose the subscribers sequentially so second in term will be S2 and then S3 so on and so forth.

The same procedure will be applied at the broadcast server end as well, but since it knows all the secret and corresponding hash functions. It has the option of doing all of this in advance and then storing them rather than calculating the keys on the fly.

Now let say that the subscribers passed the old values that were assigned to them to their respective hash functions and they get new values, we assume that they will get the following values:

Subscriber S1----------→ F1(6)=10

Subscriber S2----------→ F2(17)=18

Subscriber S3----------→ F3(34)=28

Where 'F' is the respective hash functions assigned to subscribers by the KDC. Hash function can MD5, SHA1, SHA2 etc.

Now using the above shares subscriber S1 will calculate the new group key as follows:

$a_0+a_1+ a_2 = 10$ ----------(7)

$a_0+2a_1+ 4a_2 = 18$ ----------(8)

$a_0+3a_1+ 9a_2 = 28$ ----------(9)

From 7 we get

$a_0 = 10– a_1– a_2$ ----------(10)

substituting 10 in 8 and 9 we get

$a_1 = 5$

$a_2 = 1$

putting a_1 and a_2 in 10

we get the group key which is

$a_0 = 10 - 5 - 1 = 4$

Therefore, 4 is the group key that will be used by the video server to encrypt and the subscriber will use to decrypt. The thing to notice here is that without any communication between the video server and the subscribers, the key has been changed successfully.

6.9.6 'TABLE 11' DESCRIPTION

Table 11 lists the various threats identified along with the stage of the Cloud lifecycle these threats may be active. The table also includes the classification of the threats in confidentiality, availability and integrity using the information risk rating.

Table 4 has 5 columns, the threat category column mentions the threat category that is being analysed. The threat categories are coming from the IRAM tool that was used to do the threat and vulnerability assessments. Columns 2 of the table mentions the threat itself like 'Hacking'. This column also mentions the words AIC where A stands for availability, I for Integrity and C for confidentiality. Wherever the abbreviations A is mentioned it means that the threat only relates to availability and same is true for integrity and confidentiality. The column 3 mentions the stage of the Cloud deployment whether it is operation or deployment stage. The column 4 mentions the asset involved like 'customer data'.

The Cloud deployment scenario is column 5 relates to different scenarios like bursting, federation, multi Cloud etc. Column 6 mentions the priority that is linked with the asset. Now this priority has been declared in section 5.4.9. The assets relating to confidentiality are high priority (4 or 5), assets relating to availability are medium priority (3 or 4) and assets relating to integrity would have low priority (1 or 2). Column 5 is relating to likelihood which is the possibility of risk materialising. The likelihood rating has been added by the researcher himself using his own knowledge of the domain. No metrics exist that provide likelihood ratings of Cloud computing scenarios.

There would be cases were the priority of an asset would be high because it impacts confidentiality but the likelihood of it actually materialising would be low. The risk which have high priority and high likelihood are the one which have the highest impact.

The threat numbers that are mentioned in the table are coming from the IRAM tool and therefore would be inconsistent as the researcher has omitted numerous threats which are not relevant to the Cloud computing scenarios.

6.10 CONCLUSION

This work presents how one can secure a video broadcast subscription service in the Cloud computing setup. The scalable video scenario is built on top of an IaaS Cloud and shown how the video can be encrypted and authenticated efficiently. We have also presented a secure key management protocol for the updating of encryption keys used for the encryption of the broadcast video. The key management protocol is efficient and secure - preventing a large number of attackers from breaking the security of the system. Some of these results were also presented in [157]. A number of threats that need to be monitored are identified and the assets they affect to give a risk assessment methodology of these threats. Future work will focus on calculating the time delay and efficiency affecting the setup of the video broadcasting and predicting how this will affect the performance of the video distributions process.

The above research work has been published in the IEEE International Conference for Internet Technology and Secured Transactions (ICITST-2013) [168].

Threat Category	**Threats** (video threat id) {Threat classification - Availability (A)	**Stage of Cloud** (Pre /Deploymen	**Assets involved**	**Priority** (1 is low, 5 is high)	**Likelihood** (1 is low, 5

	Confidentiality (C) Integrity (I)}	t/ Operation)			is high)
External attacks	(V1.) Carrying out of Dos (Denial of Service) attack {A}	Operation	Broadcasting server	5	4
	(V2.) Hacking {I,C}	Operation	Customer data, comprising service, company reputation	3	1
	(V3.) Undertaking malicious probes or scans {I,C}	Operation	Hypervisor code, virtual machine, video server	4	4
	(V4.) Cracking password {A,I,C}	Operation	Customer data or service	3	1
	(V5.) Cracking keys {A,I,C}	Pre-deployment, Operation	Customer data or service	2	1
	(V6.) Spoofing user identities {A,C}	Pre-deployment, Operation	Customer data or service data, all services	3	1
	(V7.) Modifying network traffic{I}	Operation	Software, connections, service, video streaming (runtime)	2	2
	(V8.) Eavesdropping {I,C}	Operation	Software, connections, service (runtime), video streaming	4	3

	(V9.) Distributing computer viruses {I}	Operation	Software, connections, service, broadcast is usually patched with security modes	2	1
	(V10.) Introducing Trojan horses {I}	Operation	Software, connections, service	3	1
	(V11.) Introducing malicious code {C}	Deployment and Operation	Software, connections, service, not through video easy to, broadcast is controlled	2	1
	(V12.) Distributing Spam{A}	Deployment, Operation	Mailing lists, server lists	2	1
Theft	(V13.) Gaining unauthorized access to systems or networks {A,I,C}	Operation	Customer data or service, extract data from the video	4	3
	(V14.) Theft of business information {A,C}	Operation	Customer data	4	2
	(V15.) Theft of computer equipment {A,C}	Pre-deployment, Operation	Customer data	1	2
System	(V16.) Malfunction of software {I}	Pre-deployment, Operation	Toolkit, all services	1	4

malfunction			video server, end-user, because of the key generation		
	(V17.) Malfunction of computer network equipment {I}	Pre-deployment, Deployment, Operation	Toolkit, all services, video server, malfunction during the key generation will affect the broadcasting of the video and the server	1	3
Service interruption	(V18.) Natural disaster {I}	Pre-deployment, Deployment, Operation	Customer data, video server	4	1
	(V19.) System overload {A,C}	Operation	Customer data, video server	1	2
Human error	(V20.) User error {C}	Pre-deployment, Deployment, /Operation	Data	3	3
System specific threat	(V21.) Data Leakage {I,C}	Operation	Data, Video data	4	2

s and abuse					
	(V22.) Data ownership {I}	Pre-deployment, Deployment	Data relates to video rights	4	2
	(V23.) Data exit rights {I,C}	Pre-deployment, Deployment	Data, SLA relating to copyrights	4	3

Table 11: Threats Identified in the Various Use Cases and their Details for Video Distributions

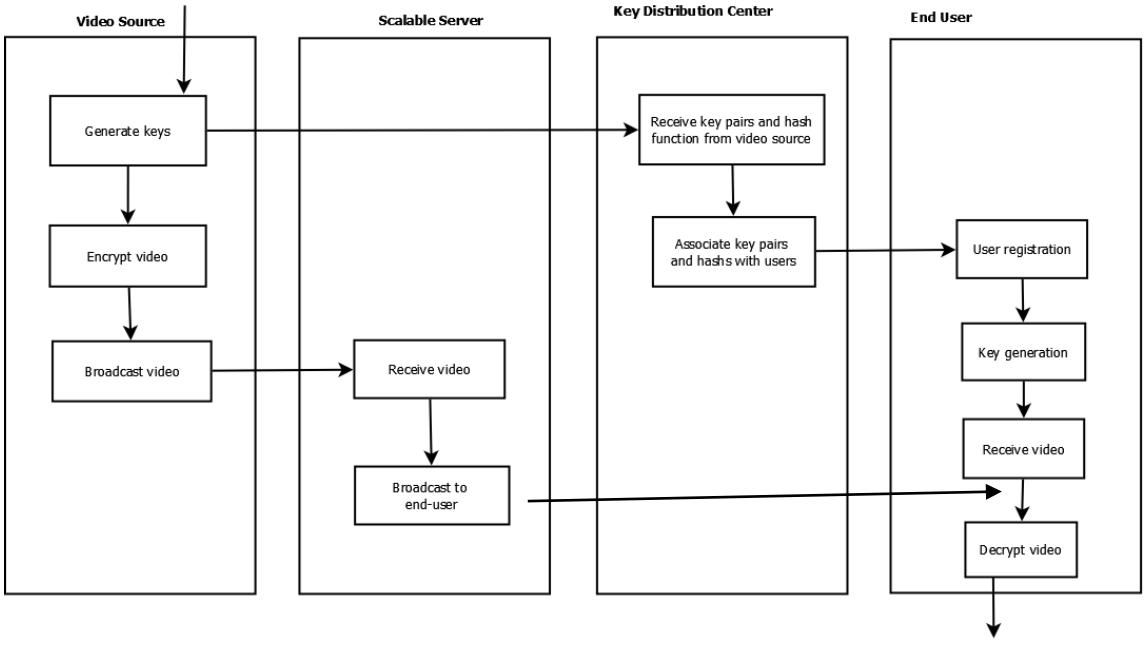

Figure 20: Flowchart Scenario

CHAPTER 7

ACCESS CONTROL AND DATA CONFIDENTIALITY IN CLOUD COMPUTING (ACDC^3)

7.1 THE SCHEME

We propose the scheme Access Control and Data Confidentiality in Cloud Computing (ACDC^3) which fills the research gaps 1 and 4, the details of the research gaps are present in Chapter 4. ACDC^3 is a scheme that guarantees confidentiality of data even when it is stored on an un-trusted third-party network while being scalable at the same time.

The fundamental idea is that we decouple the fine grained access control with data confidentiality in order to achieve scalability. These two mechanisms are thus considered separate requirements.

The explanation of the scheme is divided into two parts: (1) how scalable data confidentiality is achieved on the Cloud, (2) how we achieve scalable fine grained access on the Cloud. Note in particular that the scheme ensures that no re-encryption is required when users are revoked. The data confidentiality part is divided into two embodiments of the proposed scheme.

This research work was filed by the BT IP Department in the form of two patents to the EU Patent Office and US Patent Office. BT's reference number for the patent is Europe A32311 [169] [170].

7.2 BACKGROUND

With the advent of Cloud computing, issues of data access and data confidentiality are becoming of more and more importance. In particular the provision of secure network file storage and access control to ensure that the right users can access the right files is critical to many organisations. Whilst historically "firewall" type solutions were employed, where access control to the actual storage systems themselves was implemented, in many Cloud computing scenarios the storage systems themselves are untrusted, and it is therefore the ability to access data within such untrusted systems that is now of importance.

As described by G. Ateniese, K. Fu, M. Green, and S. Hohenberger[91] in "Improved proxy re-encryption schemes with applications to secure distributed storage," in Proc. of NDSS'05, 2005, proxy re-encryption allows a proxy to transform a ciphertext computed under Alice's public key into one that can be opened by Bob's secret key. There are many useful applications of this primitive. For instance, Alice might wish to temporarily forward encrypted email to her colleague Bob, without giving him her secret key. In this case, Alice the delegator could designate a proxy to re-encrypt her incoming mail into a format that Bob the delegatee can decrypt using his own secret key. Alice could simply provide her secret key to the proxy, but this requires an unrealistic level of trust in the proxy. Instead, therefore, Alice computes a re-encryption key from Bob's public key, the re-

encryption key being a function that converts incoming mail intended for Alice and encrypted with her public key into a form that permits decryption by Bob's private key. Alice then provides the re-encryption key to the proxy, which re-encrypts the incoming mail, and passes it to Bob. Bob can then decrypt the mail intended for Alice with his private key.

Several proxy re-encryption schemes are described in the Ateniese paper, specifically section 3 thereof, any details of which necessary for understanding the present invention being incorporated herein by reference. Ateniese et al also comment that proxy re-encryption has many exciting applications in addition to previous proposals for email forwarding, law enforcement, and performing cryptographic operations on storage-limited devices. In particular, according to Ateniese et al. proxy cryptography has application to secure network file storage, and they describe a specific file system which uses an untrusted access control server to manage access to encrypted files stored on distributed, untrusted block stores, and that uses proxy re-encryption to allow for access control without granting full decryption rights to the access control server.

In the Ateniese file system, end users on client machines wish to obtain access to integrity-protected, confidential content. A content owner publishes encrypted content in the form of a many-reader, single writer file system. The owner encrypts blocks of content with unique, symmetric content keys. A content key is then encrypted with an asymmetric master key to form a lockbox. The lockbox resides with the block it protects.

Untrusted block stores then make the encrypted content available to everyone. Users download the encrypted content from a block store, then communicate with an access control server to decrypt the lockboxes protecting the content. The content owner selects which users should have access to the content and gives the appropriate delegation rights to the access control server.

The content keys used to encrypt files are themselves securely encrypted under a master public key, using a unidirectional proxy re-encryption scheme of the form described in the Ateniese paper. Because the access control server does not possess the

corresponding secret key, it cannot be corrupted so as to gain access to the content keys necessary to access encrypted files. The secret master secret key remains offline, in the care of a content owner who uses it only to generate the re-encryption keys used by the access control server. When an authorized user requests access to a file, the access control server uses proxy re-encryption to directly re-encrypt the appropriate content key(s) from the master public key to the user's public key.

Operation of the proxy re-encryption file system of Ateniese is shown further in Figure 21. Here, the user's client machine fetches encrypted blocks from the block store. Each block includes a lockbox encrypted under the master public key. The client then transmits lockboxes to the access control server for re-encryption under the user's public key. If the access control server possesses the necessary re-encryption key, it re-encrypts the lockbox and returns the new ciphertext. The client can then decrypt the re-encrypted block with the user's secret key, to obtain the symmetric content key encrypted therein. The symmetric content key is then used to decrypt the content of the data block.

Ateniese et al. therefore provide an access control server storage scheme where much of the security relies on the strength of a provably-secure cryptosystem, rather than on the trust of a server operator for mediating access control. Because the access control server cannot successfully re-encrypt a file key to a user without possessing a valid delegation key, the access control server cannot be made to divulge file keys to a user who has not been specifically authorized by the content owner, unless this attacker has previously stolen a legitimate user's secret key.

However, Ateniese et al. take absolutely no account of the issue of revocation of user access rights to the data. In their scheme, the symmetric content key that is used to encrypt the data stored in the block store is passed to the end user, via the proxy re-encrypted lock box. Once the end user has obtained the symmetric encryption key, it can then continue to access the data in the block store encrypted with this key (because the block store itself has no access control). In order to prevent this access it would be necessary to re-encrypt the data in the block store. However, in this respect in typical

Cloud computing scenarios there would be numerous infrastructure providers all providing services to millions of data consumers. It is simply not possible to re-encrypt data every-time a user has his or her access revoked. This is because there would be many data consumers who would be having their access revoked in a very short span of time, and hence there would need to be more than one re-encryption operation taking place at once. It would therefore be very hard if not impossible to keep track of which data was encrypted with which key.

In view of the above, there is a still a clear need to provide data access control schemes for network stored data which are able to effectively control data access whilst taking into account the possibility for user access rights to be revoked.

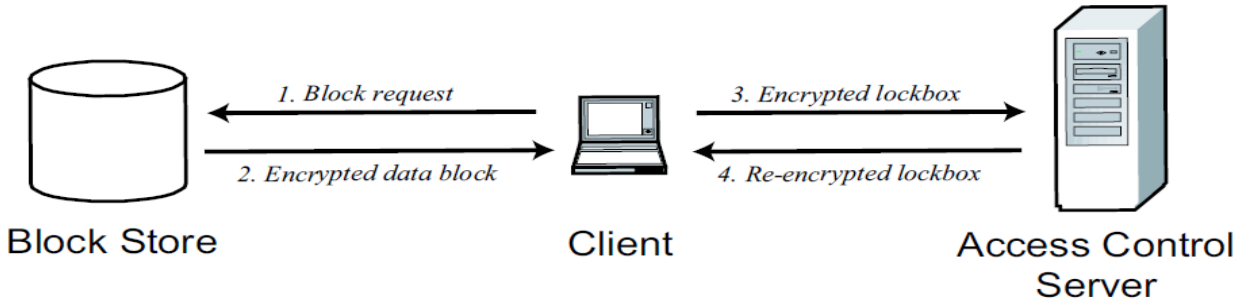

Figure 21: Prior Art

7.3 SCENARIO

The scenario can be better understood by taking into account the example of medical records being hosted on the infrastructure provider. In this example a medical centre would be the data owner whereas the data consumers are the patients, nurses or doctors accessing the data. The medical centre is using the infrastructure provider to host data on the Cloud. According to the health insurance and portability act (HIPPA) [53] it is a requirement for all medical centres to ensure the confidentiality of data when the data is hosted on infrastructure provider. Furthermore commercial offerings such as the Amazon S3 and the GoogleDocs cannot be trusted with data that is of commercial or of

confidential nature. In this scenario data can only be hosted on these infrastructure providers when appropriate confidentiality and authorisation controls are there.

This section presents the parties involved in the protocol. The system is composed of the following parties:

- Data Owner
- Data Consumers
- Infrastructure provider
- Trusted authority

The data owner owns the files stored at the infrastructure provider. The Data owner is responsible for encrypting these files. The data owner resumes control over the VM or the machine that is hosting the trusted authority by control we mean that the data owner is the only one that has administrative level access over the operating system. The physical infrastructure may be controlled by the provider but as long as the machine that is hosting the trusted authority is not compromised then the scheme is secure. The data owner has full read and write access on the files stored at the infrastructure provider.

The scenario has two main parts. The first part is when the data owner wants to transfer data to the infrastructure provider. The second part is that of when data consumer wants to access data hosted on the infrastructure provider.

In the first part, the following steps are taken by the data owner,

1. The data owner encrypts data using keys which are only known to him at that time.
2. The data owner then transfers data to the infrastructure provider.
3. The encryption that is done by the data owner is one time encryption and there is no need for the data owner to update data when keys or users get revoked.
4. The data owner is also responsible for the issuance of re-encryption keys to the trusted authority.
5. The data owner also sends the final part of the key to the data consumer.

In the second part the data consumer takes the following steps,

1. The data consumer sends a request to the trusted authority that it wants to access data on the infrastructure provider.
2. The infrastructure provider then performs access control checks.
3. If permission is granted to access the file stored on the infrastructure provider then the trusted authority sends request to the infrastructure provider to fetch the file. After receiving the encrypted file, the trusted authority then performs re-encryption of the file using re-encryption keys given to it by the data owner. The trusted authority uses a proxy re-encryption scheme (see Ateniese) to achieve the objective of data confidentiality on the Cloud. The trusted authority could reside at the infrastructure provider where the data is hosted or at another infrastructure provider. It could also be an independent entity in the scenario. The trusted authority requires substantial computing power as it would perform the re-encryption of data.
4. The trusted authority then forwards the file to the data consumer.
5. The data consumer then sends request to the data owner to fetch the final part of the key.
6. After receiving the key from the data owner the data consumer then performs the decryption of the file.

In figure 18, the architecture of the scheme ACDC[3] is shown.

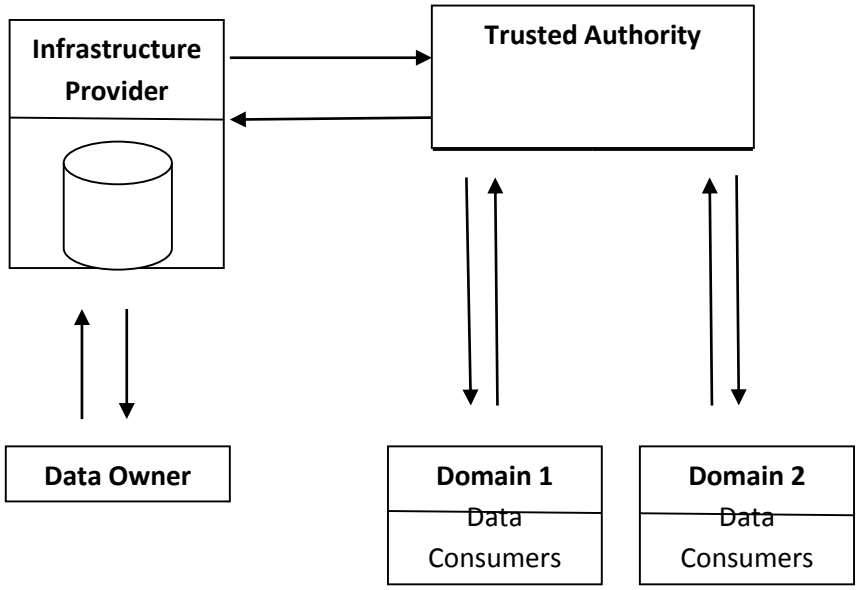

Figure 22: Scenario of ACDC[3]

7.4 EMBODIMENTS OF THE SCHEME

Two embodiments of the invention will now be described. In both the embodiments' data is stored in an encrypted form in a network storage facility, by a data owner. In order to allow access to the data by a third party for instance the data consumer, it is stored at the network storage facility (Infrastructure Provider). The proxy re-encryption of the data stored in the network storage facility is done by a trusted authority. This operation converts the data into a form where it can eventually be decrypted by the data consumer. However, the protocols of each embodiment are such that without the proxy re-encryption by the trusted authority it would not be possible for the data consumer to decrypt data obtained directly from the network storage facility, even if having been previously provided with a decryption key from a previous operation. This therefore allows for access control to be administered by the trusted authority, and for user access

rights to thereby be revoked without the user still being able to access and decrypt to plaintext data stored in the network storage facility.

In more detail, the data stored at the network storage facility is encrypted with one or more layers of encryption, one of which is an asymmetric encryption layer using the public key of the data owner. In order to allow this layer to be removed, the data owner provides a trusted authority with a re-encryption key, to re-encrypt the data so that the data owner public key encryption layer may be removed. The target of the re-encryption may be the requesting data consumer (for example where the data owner public key encryption layer is the only encryption applied to the data) in which case the re-encrypted data may be passed to the data consumer, who then decrypts it with his private key. Alternatively, where more than one encryption layer is used with the data (for example, a symmetric encryption, followed by the data owner public key encryption), then the target of the re-encryption may be the trusted authority itself, wherein the asymmetric public key encryption layer may be removed by the trusted authority by re-encrypting the data using a re-encryption key generated by the data owner for the trusted authority, and then decrypting using the trusted authority's private key. In both cases the data consumer only gets access to the data via the trusted authority, which must undertake the re-encryption, without which the data consumer is unable to access plaintext data.

Both embodiments of the invention are based on the same system architecture, shown in Figure 22.

Figure 23 illustrates a typical system configuration of one of the actors in the architecture of the embodiments. In this respect, each "actor" will typically be provided with a processor based communications device, such as a general purpose computer such as a laptop or desktop, or other communications device such as a smartphone, tablet, set-top box, games console, or the like. Within Figure 19 any such processor based communications device is provided with a CPU, memory, one or more input/output interfaces (such as video and audio output controllers, as well as user input device

controllers such as any one or more of a keyboard, touchscreen, or mouse controller, for example) and one or more network interfaces (such as one or more wired or wireless network adapters, for example). In addition it is provided a storage medium such as a hard disk, flash drive, or other (usually non-volatile) data storage on which is stored the system operating system, as well as a data access control program, that acts to control the system to operate according to the communications and security protocols of the embodiments of the invention, to be described. Also provided is a web browser program, which when run allows the system user to browse the World Wide Web. In this regard, the computer system communicates via the network interface with one or more remote servers or other devices, via a network such as the Internet or an intranet. Other programs and for other purposes may of course also reside on the same computer readable medium.

As noted, the data access control program enables the device to operate according to its role in the present architecture as one of the actors, and to implements the security and communications protocols to be described in respect of each of the embodiments. Therefore, where the device is acting as a data consumer then the program controls the device to perform the actions of a data consumer, to be described. Likewise, when the device is a data owner, or a trusted authority, the program controls the device to perform the respective actions of each actor, as required. Of course, the program need not be a single computer program, and may be a suite of programs that work together. Likewise, any device which is participating as an actor need only have those programs or part of a program that cause it to fulfil its necessary actions under the protocols of the embodiments.

In addition to the above, in both embodiments to be described there are seven main security requirements and assumptions involving the following issues: collusion resistance, access control, data channels, data confidentiality, read/write requests, trusted authority and management of keys.

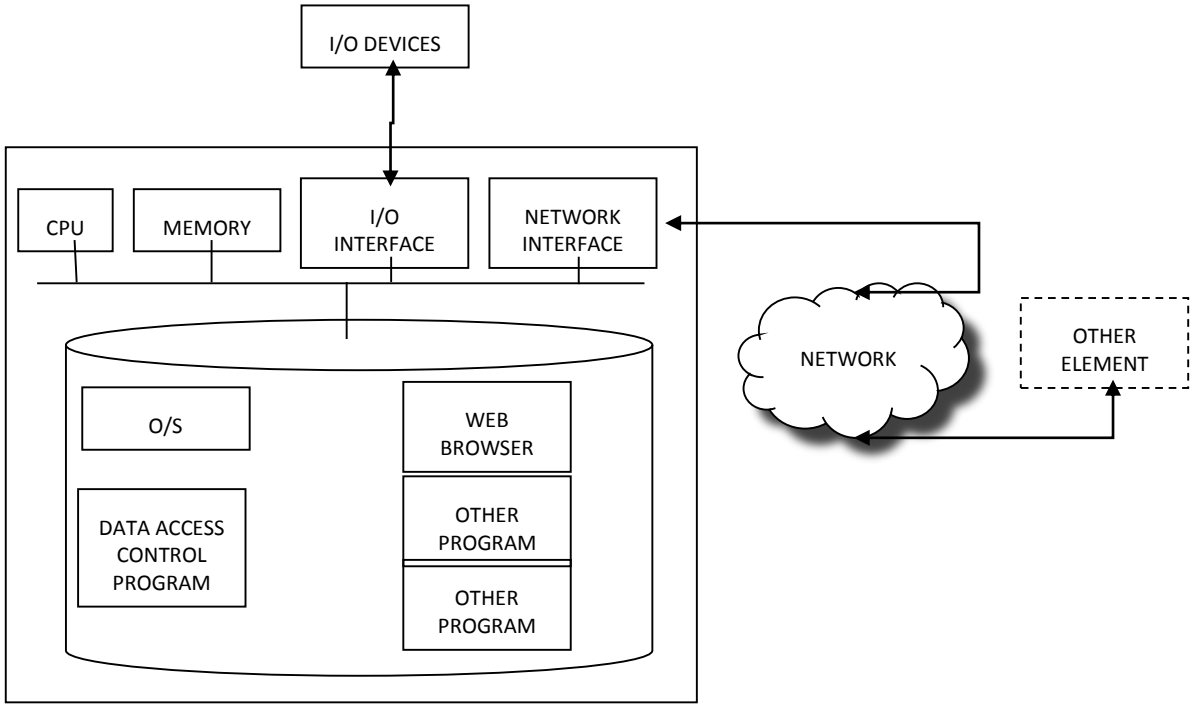

Figure 23: Device Level Architecture of the Scheme

7.5 ASSUMPTIONS AND SECURITY REQUIREMENTS

In this section, we present the 7 main security requirements and assumptions about the following points: collusion resistance, access control, data channels, data confidentiality, Read/Write requests, trusted authority, and management of keys.

Collusion resistance: The scheme should ensure that data consumers should not be able to decrypt the encrypted data even when colluding with the infrastructure provider. Contrary to the assumption made in other schemes[171], [172] and [92], we do not consider that the infrastructure provider is curious but honest because this assumption does not hold in the Cloud computing scenario previously presented. In our scheme the assumption is that the infrastructure provider does not restrict itself for decrypting data or finding information about the access control policies.

Access Control: The scheme ensures that data consumers bearing the correct attributes are able to access the data. Unauthorised data consumers who do not have the right

attributes should be prevented from accessing the data. Even if the infrastructure provider colludes with the data consumers they should not be able to access data they are not allowed to read.

Data Channel: We assume that all data channels that exist between the actors in the scenario are secured. The network level security is outside the scope of this work.

Data Confidentiality: The scheme should ensure backward and forward secrecy. In backward secrecy any data consumer who accesses files should not be able to decrypt files exchanged in previous communications with another data consumer. In forward secrecy a data consumer should not be able to decrypt files using old credentials to decrypt files exchanged in subsequent communication.

Read/Write Request: In the scheme, we make the assumption that data consumers would only make read requests. Any write request would only be made by the data owner or it would come via the data owner.

Trusted Authority: The trusted authority has considerable computational power available to process requests coming in. The assumption is that there should not be any bottleneck created by the trusted authority by not being able to process incoming requests. We also envision that the trusted authority could reside on the premises of the data owner, or on the premises of the infrastructure provider. As long as the machines on which the trusted authority is running is not compromised then the scheme is secure.

Management of Keys: The exchange of keys between the actors of the scenario is not part of the scheme. We assume that there is a baseline level of trust that exists between the actors and they are able to exchange the keys and update them appropriately.

Local content: We do understand that even a user that has its access revoked would be able to access the local copy of the data that he has already downloaded.

7.6 SCHEME DESCRIPTION (EMBODIMENT 1)

The first embodiment comprises two phases, a data storage phase, and a data access phase. The actors of the first embodiment are those described previously with respect to Figure 23 i.e. a data owner, an infrastructure provider, a trusted authority and a data consumer.

The data storage phase is shown in Figure 24. Here, a data owner first generates the public private key pair required for asymmetric encryption. Then, the data owner encrypts the data to be stored with his public key i.e. CT = E(DATA, DO-PubK), and uploads the encrypted data CT to the infrastructure provider. The infrastructure provider task is to store the encrypted data. This concludes the data storage phase, which may be repeated as many times as necessary for different files, or different blocks of data. In this respect, however, it is not necessary for the data owner to generate a new public-private key pair per file or data block, and the same key pair may be used for several files or blocks.

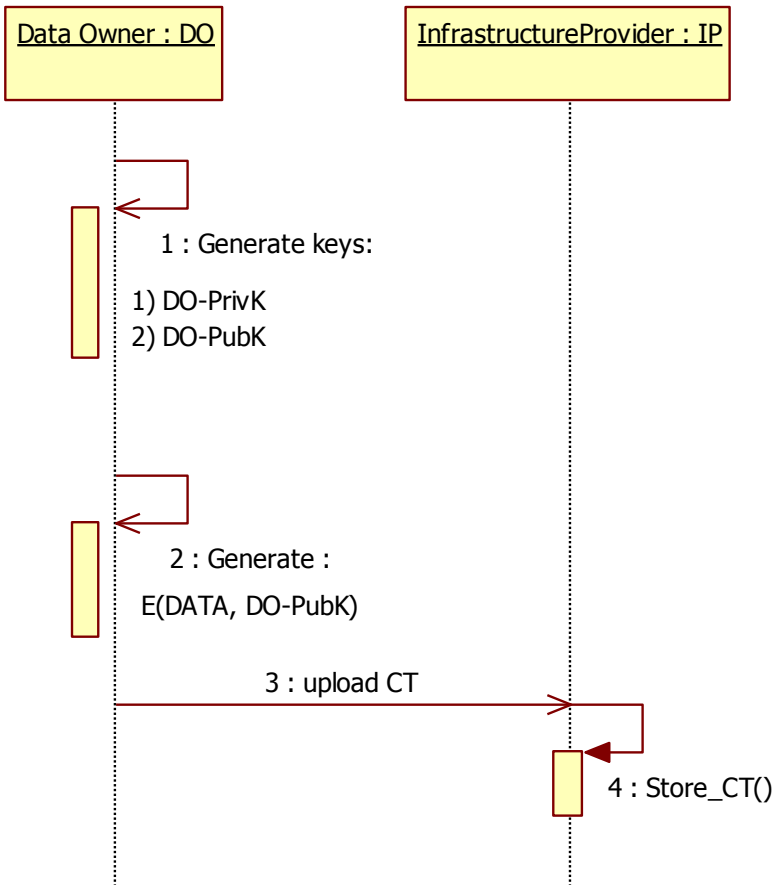

Figure 24: Environment Setup (Embodiment 1)

The data access phase is shown in Figure 25. Here, the data consumer (DC) transmits a data access request to the trusted authority (TA), identifying himself and specifying which data he wishes to access. In addition, in this embodiment the data consumer also passes as part of the data access request a request token, comprising the data consumer's private key encrypted with the data owner's public key. This is required in this embodiment for the data owner (DO) to generate a re-encryption key with the target as the requesting data consumer, as will become apparent below.

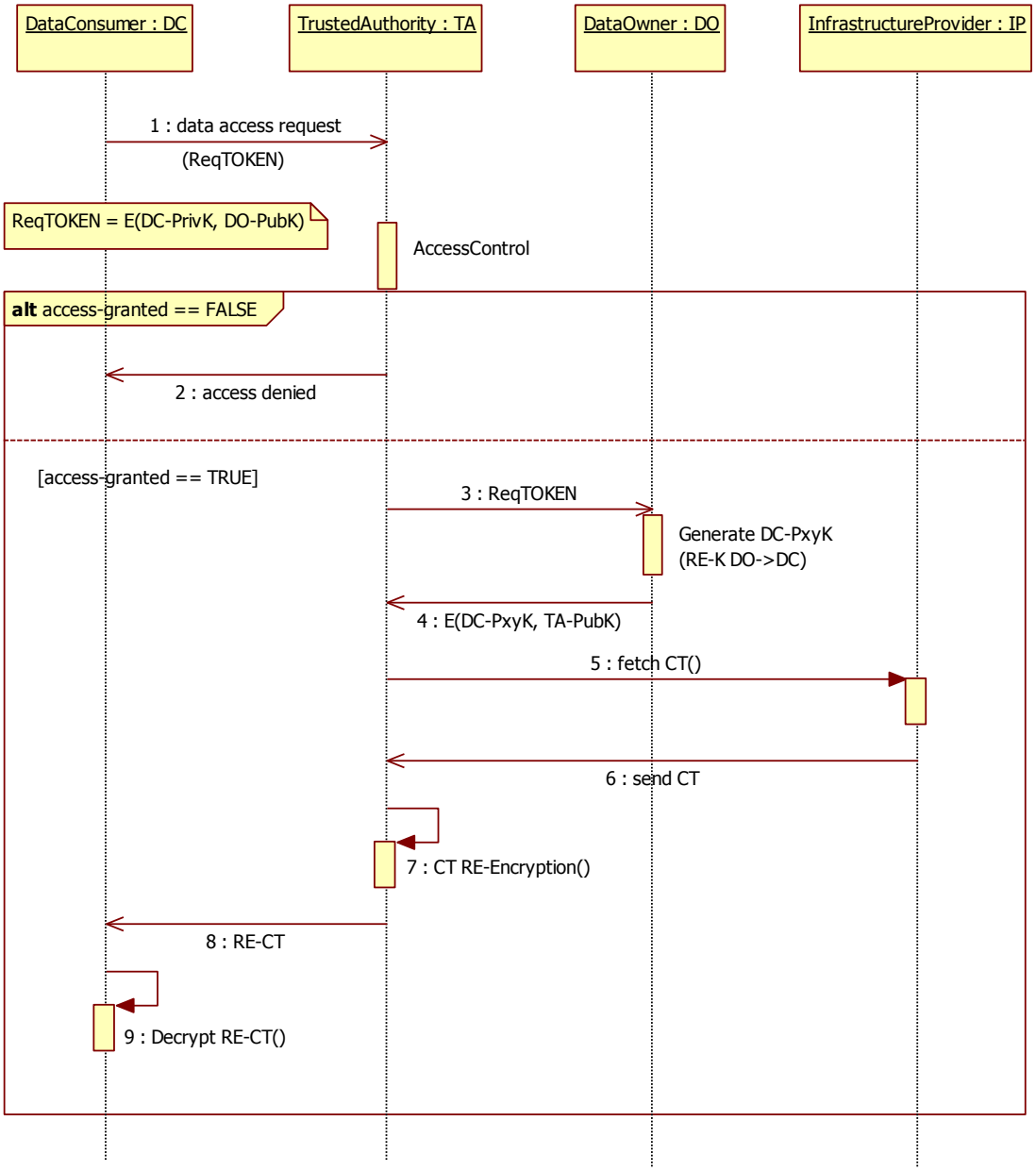

Figure 25: Data Access (Embodiment 1)

The trusted authority (TA) then undertakes an access control procedure, where it determines whether the requesting data consumer (DC) is an authorised person to access the data, for example by consulting a list or other database containing the identities of authorised users. If the trusted authority determines that the data consumer is not authorised then an “access denied” message is passed back to the requesting data

consumer, and the data access phase then ends. However, if the TA determines that the DC has access rights then the request token received from the DC is passed to the data owner (DO). The DO then decrypts the request token with his own private key to obtain the DC's private key, and then generates a proxy re-encryption key DC-PxyK for the data consumer, which is a function which transforms data encrypted with the DO's public key into data that can be decrypted with the DC's private key. The re-encryption key DC-PxyK is then encrypted with the TA's public key, and sent to the TA.

The TA therefore at this point in time has received a request to access a particular data file or block from the DC, and has granted the request. It has also received from the DO a proxy re-encryption key which will be able to re-encrypt data encrypted with the DO's public key into data that can be decrypted with the DC's private key. Afterwards the TA requested data CT from the infrastructure provider, which is done by a request-response mechanism. The TA therefore receives CT from the infrastructure provider. Recall that CT is encrypted with the DO's public key.

In order to allow the encryption layer to be removed by the DC, the TA uses the proxy re-encryption key it received from the DO to re-encrypt CT. After the re-encryption CT remains encrypted, as Re-CT, and hence cannot be read by the TA, or any other actor other than the DC the target of the re-encryption (including malicious eavesdroppers). However, Re-CT can be decrypted by the DC using its private key. Re-CT is sent by the TA to the DC, where it is then decrypted using the DC private key. The decryption of Re-CT at the DC ends the data access phase.

7.6.1 ACCESS MATRIX

In order to analyse the above protocol, in the following Figure, we introduce a symbolic 3-way representation in order to easily summarize all the information in an access matrix developed from the protocol. The table 12 below explains each block A, B, or C.

Symbol	**Meaning**	**Values**

A	Can the entity obtain directly this information?	• "◊" : YES because the entity generates this data • "Y" : YES • "N" : NO
B	If the data is encrypted, what is needed to decrypt it ?	One or more keys
C	Which info can be decrypted ?	• Data, CT, CT', CT'' • "-" : No one because is not possible to access the info in the block B

Table 12: Access Matrix (Embodiment 1)

Blocks B and C are optional and appear only if block A is "Y". The access matrix can be organized as follows: shown on the rows are the entities involved in the process, and shown on the columns are each transactional state of the *data*. Each entry therefore contains a 3-way block, or alternatively only its part A. Figure 26 shows the access matrix thus derived for the first embodiment.

Entity	CT	RE-CT
DO (Data Owner)	◊	-
IP (Infrastructure Provider)	Y: $DO_{Priv\text{-}KEY}$ -	N
STA (Semi-Trusted Authority)	Y: $DO_{Priv\text{-}KEY}$ -	Y: $DC_{Priv\text{-}KEY}$ -
DC (Data Consumer)	N	Y: $DC_{Priv\text{-}KEY}$ -
MA (Malicious)	Y: $DO_{Priv\text{-}KEY}$ -	Y: $DC_{Priv\text{-}KEY}$ -

Figure 26: Access Matrix (Embodiment 1)

From Figure 22 we can see that in order to access CT then the private key of the data owner is always required, whereas for Re-CT the private key of the data consumer is required. Therefore, if a malicious eavesdropper intercepts communications between the parties they will not be able to access any data, as they will have neither private key. Likewise, the data consumer can only ever access re-encrypted data, that has been re-encrypted so as to be decrypted with the data consumer's private key. This allows for user revocation by controlling access rights of users at the trusted authority, in that the trusted authority will only re-encrypt for a user that is authorised. Once authorisation has been lost for a user at the trusted authority, then no re-encryption will occur. Even if the data consumer then colludes with the infrastructure provider to access the data, he will

not be able to decrypt the data because the data the infrastructure provider stores i.e. CT requires decryption with the data owner's private key only.

One drawback of the first embodiment as described above is that the data consumer sends a token in which its private key is encrypted using the public key of the data owner. This token is only forwarded to the data owner if the data consumer is given access permission by the access control mechanism in the TA. However, sharing of the private key is not feasible in many scenarios where the data consumer wants to keep full control over its private keys. In order to get around this issue, therefore, we present the second embodiment of the scheme. Moreover, the data encryption in this embodiment is done using asymmetric encryption. When DO encrypts data and then transfers it to the IP, although it is a one off operation but requires significant computational overhead. Therefore the requirement is there to develop an embodiment which uses symmetric encryption for data encryption rather than asymmetric encryption. Asymmetric encryption is 1000 times slower than symmetric encryption[173][174].

In order to solve the above two issues we present to you the embodiment 2 of the scheme. The embodiment 2 will be used as the standard embodiment for the rest of the thesis.

7.7 SCHEME DESCRIPTION (EMBODIMENT 2)

In this section we provide the description of the embodiment 2 of the scheme. The operations conducted by the embodiment 2 are the following ones:

Key generation: At the Data owner end it has to generate a symmetric key and public/private key pair. Also it is responsible for generating the re-encryption keys for the trusted authority.

At the Data Owner end following keys have to be generated:

- DO_{SK} : Data Owner Symmetric key
- DO_{PK} : Data Owner Public Key

- DO_{PR} : Data Owner Private Key

At the Trusted authority end, the following keys have to be generated:

- TA_{PK}: Trusted Authority Public Key
- TA_{PR}: Trusted Authority Private Key

At the Data Consumer end, the following keys have to be generated:

- DC_{PK}: Data Consumer Public Key
- DC_{PR}: Data Consumer Private Key

Re-encryption key generation: The data owner also generates a re-encryption key per trusted authority. The data owner uses the DO_{PR} and the TA_{PK} to generate the re-encryption key for each specific trusted authority TA. We use the following symbol for the key

- RK_{TA}: Re-encryption key

Core Encryption: The core encryption is the process of transforming plain text into cipher text by using the DO_{SK} by the data owner. The cipher text is now called DO_{SK}(Text).

Second level encryption: Second level encryption is done using the DO_{PK} by the data owner. This data can only be decrypted using the DO_{PR} of the data owner or the delegates re-encryption keys. Now the new cipher text is proxy ready and is also ready to be delegated to the trusted authorities. The cipher text here is now called CT'.

First level encryption: First level encryption is the process of converting CT' to CT. It includes two sub-processes, firstly the trusted authority uses the re-encryption key RK_{TA} to convert the CT' to CT''. Secondly it uses the TA_{PR} to convert the CT'' to CT.

Decryption: Decryption is performed by the data consumer using the symmetric key DO_{SK} that the data owner has provided to it. The key is provided to the data consumer by using its public key DC_{PK} to decrypt the symmetric key DO_{Sk}.

a. Environment Setup

In figure 27, the environment setup of embodiment 2 is presented, following are the steps,

1. The data owner performs the core encryption of data by using DO_{SK}(Text) = CT.
2. In the second step, the CT is transformed into CT' = DO_{PK} (DO_{SK}(Text)).
3. The data is now proxy ready and is now hosted on the infrastructure provider.
4. The data owner now generates re-encryption key per trusted authority.(TA-1 … TA-N)

b. Data Access

In figure 28, the data access of embodiment 2 is presented, following are the steps,

1. In the first step the data consumer makes the request to the trusted authority to access a file.
2. At the trusted authority the access control component performs fine grained access control on the request.
3. If the access control component gives permit to the request then the trusted authority sends request to the infrastructure provider to fetch the appropriate file CT'.
4. Now the trusted authority data confidentiality component performs re-encryption of the file using the re-encryption key given to it by the data owner. This will transform the CT' to CT''. This is the first level encryption refer to section 5 for more detail.
5. Now the data confidentiality component performs proxy decryption that transforms the CT'' to CT.
6. Now the trusted authority forwards the CT to the data consumer.
7. The data consumer now requests the data owner to send the DO_{SK}. The data owner using DC_{PK} encrypts the DO_{SK} and sends it to the data consumer.

8. Using the DO_{SK} the data consumer then decrypts CT to plain text.

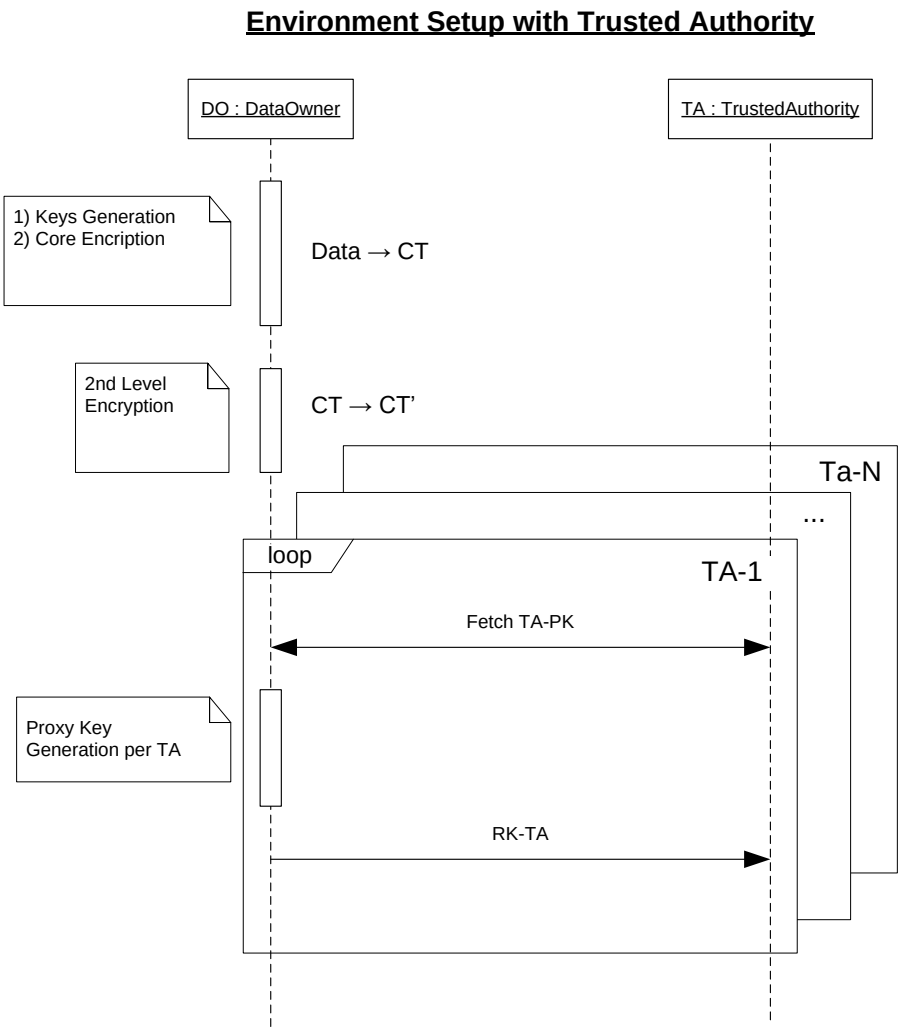

Figure 27: Environment Setup (Embodiment 2)

Data Access

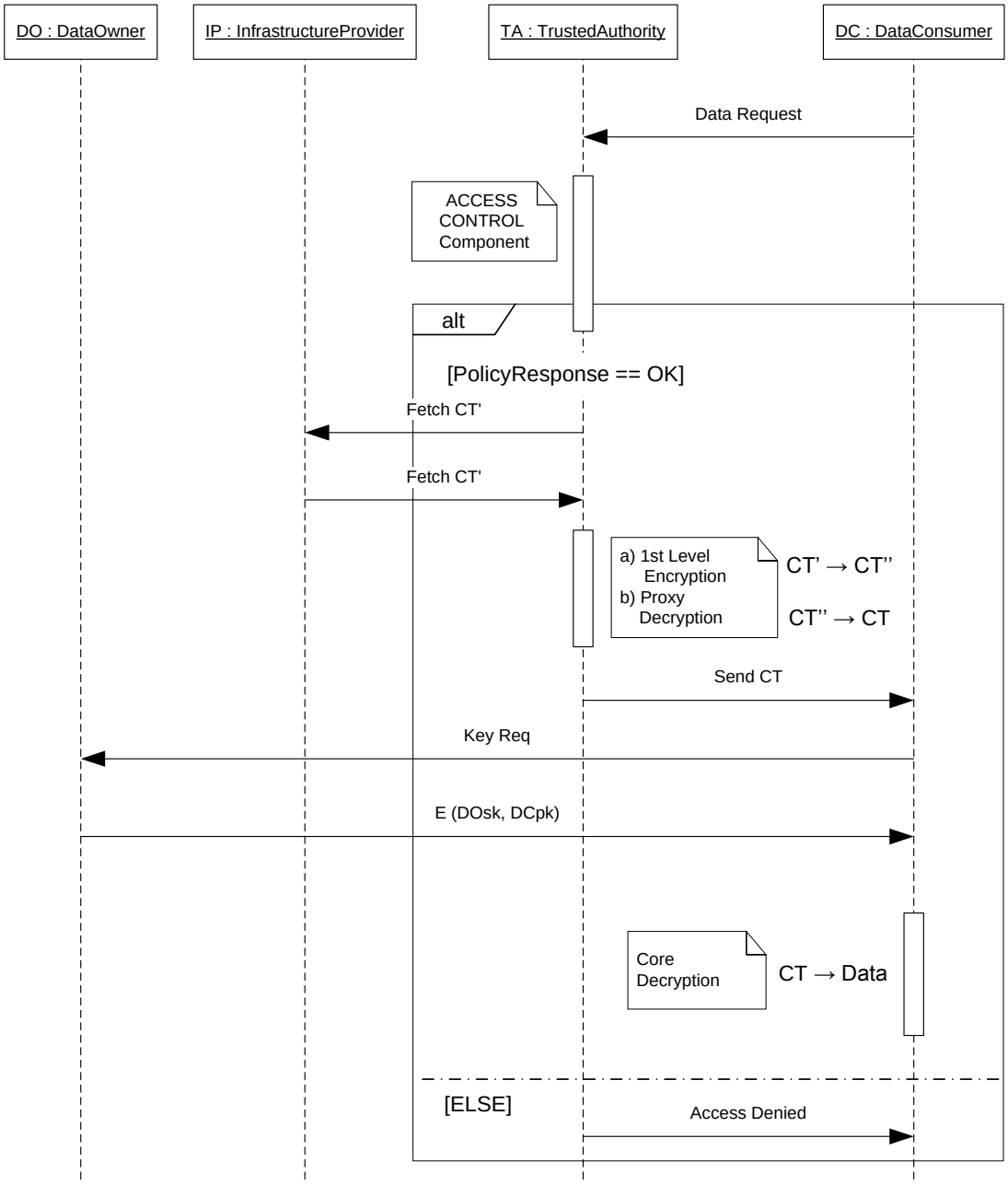

Figure 28: Data Access (Embodiment 2)

7.7.1 MATHEMATICAL FORMALISATION

The fundamental concept used in developing the Ateniese proxy cryptography scheme is that of bilinear maps. In this section the mathematical formalisation are based on the Ateniese scheme [23].

Let G_1, G_2, G_3 be cyclic groups of the prime order q.

Function e: $G_1 \times G_2 \rightarrow G_3$ is a bilinear map if for all $g_1 \in G_1$, $g_2 \in G_2$, a, b $\in \mathbb{Z}_q$, that $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$

The algorithm uses bilinear maps of the form of e: $G_1 \times G_1 \rightarrow G_2$ where $G_1 = \langle g \rangle$. e must be efficiently computable. Also, e must be non degenerate; that is $\langle e(g,g) \rangle \in G_2$

The whole process is composed of a tuple of (possibly probabilistic) polynomial time algorithms KG, RG, $\vec{E}$, R, $\vec{D}$

Key generation (KG)

$\langle g \rangle = G_1$ of prime order q

$SK_a = a \in \mathbb{Z}_q^*$ randomly selected.

$SK_b = b \in \mathbb{Z}_q^*$, randomly selected.

$PK_b = g^b$, $PK_a = g^a$, random $r \in \mathbb{Z}_q^*$

$Z = e(g,g)$

That means on *input* of a generator g, the KG algorithm *outputs* a couple of tuples (PK_a, SK_a) and (PK_b, SK_b).

Re-encryption Key generation (RG)

$RK_{A->B} = (g^b)^{1/a} = g^{b/a}$

On *input* of (PK_a, PK_b), the re-encryption key generation algorithm RG *outputs* a key $RK_{A->B}$ for the proxy.

Encryption

$m \in G_2$

$C_a = (Z^r \cdot m, g^{ra})$

On *input* of PK_a and a message $m \in G_2$, for all $E_i \in \vec{E}$ the *output* is a ciphertext C_a

Re-encryption

$C_a = (Z^r \cdot m, g^{ra})$

$C_b = (Z^r \cdot m, e(g^{ra}, RK_{A->B}))$

$= (Z^r \cdot m, e(g^{ra}, g^{b/a}))$

$= (Z^r \cdot m, Z^{rb})$

On input of $RK_{A->B}$ and a ciphertext C_a, the re-encryption function R outputs C_b.

Decryption

(Alice)

$$m = \frac{Z^r \cdot m}{e\left(g^{ra}, g^{1/a}\right)} = \frac{Zr \cdot m}{Zr}$$

On input of SK_a and a ciphertext C_a, then exists a $D_i \in \vec{D}$ that outputs the message $m \in G_2$

(Bob)

$$m = \frac{Z^r \cdot m}{\left(Z^{rb}\right)^{1/b}}$$

On input of SK_b and a ciphertext C_b, then exists a $D_i \in \vec{D}$ that outputs the message $m \in G_2$

More formally, let key pairs (PK_a, SK_a) and (PK_b, SK_b), be generated according to KG, belong to parties A and B, respectively, and let $RK_{A->B}$ be generated according to RG. Then, for all messages *m* in the space G_2, the following equations hold with probably one :

$$\forall E_i \in \vec{E}, \exists D_j \in \vec{D}, \;\; D_j\big(SK_A, E_i(PK_A, m)\big) = m \text{ for Alice}$$

$$\forall E_i \in \vec{E}, \exists D_j \in \vec{D}, \;\; D_j\Big(SK_B, R\big(\mathrm{RK_{A\rightarrow B}}, \mathrm{E_i}(PK_A, m)\big)\Big) = m \text{ for Bob}$$

In our specific scenario, skipping the key generation process already shown in §4 :

Core encryption : $CT = E\ (Data, DO_{SK})$

2nd Level Encryption : $CT' = E\ (CT, \quad DO_{PK})$

Re-encryption

$C_a = (Z^r \cdot m, g^{ra})$

$C_b = (Z^r \cdot m, e(g^{ra}, RK_{A->B}))$

$= (Z^r \cdot m, e(g^{ra}, g^{b/a}))$

$= (Z^r \cdot m, Z^{rb})$

On input of $RK_{A->B}$ and a ciphertext C_a, the re-encryption function R outputs C_b.

Decryption

(Alice)

$$m = \frac{\boldsymbol{Z^r}.\ \boldsymbol{m}}{e\left(g^{ra}, g^{1/a}\right)} = \frac{\boldsymbol{Zr}\ .\ \boldsymbol{m}}{\boldsymbol{Zr}}$$

On input of SK_a and a ciphertext C_a, then exists a $D_i \in \vec{D}$ that outputs the message $m \in G_2$

(Bob)

$$m = \frac{\boldsymbol{Z^r}\ .\ \boldsymbol{m}}{\left(Z^{rb}\right)^{1/b}}$$

On input of SK_b and a ciphertext C_b, then exists a $D_i \in \vec{D}$ that outputs the message $m \in G_2$

More formally, let key pairs (PK_a, SK_a) and (PK_b, SK_b), be generated according to KG, belong to parties A and B, respectively, and let $RK_{A->B}$ be generated according to RG. Then, for all messages *m* in the space G_2, the following equations hold with probably one :

$$\forall E_i \in \vec{E}, \exists D_j \in \vec{D},\ \ D_j\left(SK_A, E_i(PK_A, m)\right) = m \text{ for Alice}$$

$$\forall E_i \in \vec{E}, \exists D_j \in \vec{D},\ \ D_j\left(SK_B, R\left(\mathrm{RK_{A\to B}}, \mathrm{E_i}(PK_A, m)\right)\right) = m \text{ for Bob}$$

In our specific scenario, skipping the key generation process already shown in §4 :

Core encryption : $CT = E\ (Data, DO_{SK})$

2nd Level Encryption : $CT' = E\ (CT,\ \ \ DO_{PK})$

1st Level Encryption :

CT'' = R (CT', RK_{TA})

CT = D (CT'', TA_{PR})

Decryption: DATA = $D(CT,D(E(DO_{SK},DC_{PK}),DC_{PR}))$

7.7.2 ACCESS MATRIX

In figure 29, the *data* lifecycle is shown, which is an iterative cascade model.

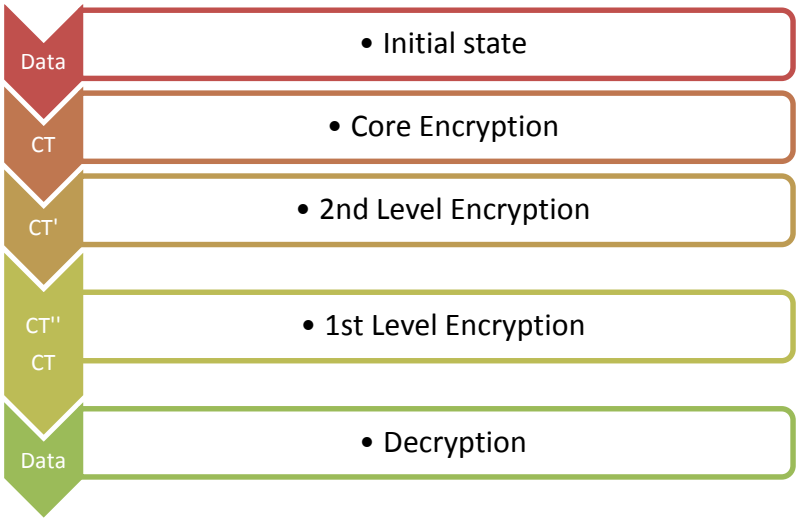

Figure 29: Data life cycle (Embodiment 2)

In each of these states, the *data* is represented by the formal statement described in the previous section. In order to define the granularity of protection mechanisms, a so called *Access Matrix* can be used as formalization for the static access permission in any step of interaction between all the entities of our scenario (Data Owner, Infrastructure Provider, Trusted Authority, Data Consumer and a Maliciuos user).

This simple formalization does not model the rules by which permission are setted in the system, but the way each party can access the *data*, taking into consideration the system's access control security policies. We introduce a symbolic 3-way representation in order to easily summarize all this information in each entry of the matrix.

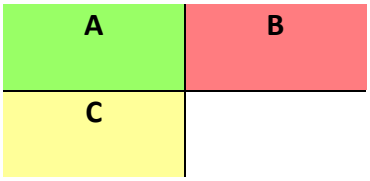

Table 13 explains each coloured block.

Symbol	Meaning	Values
A	Can the entity access the data ?	• "◊" : YES because the entity generates this data • "Y" : YES • "N" : NO
B	If the data is encrypted, what is needed to decrypt it ?	One or more keys
C	Which info can be decrypted ?	• Data, CT, CT', CT'' • "-" : No one because is not possible to access the info in the block B

Table 13: Symbol, Meaning and Values

The access matrix is organized in Table 14. On the rows the entities involved in the process, on the columns each transactional state of the *data*, each entry contains a 3-way block or only its part A.

Entity	Data	CT	CT’	CT’’
DO (Data Owner)	◊	◊	◊	N
IP (Infrastructure Provider)	N	N	Y | RK_{TA} SK_{TA} -	N
TA (Trusted Authority)	N	Y | DO_{SK} -	Y | CT’’ -	◊ | RK_{TA} SK_{TA} -
DC (Data Consumer)	Y	Y | DO_{SK} Data	N	N
MA (Malicious)	Y | DO_{SK}	Y | RK_{TA}	Y | RK_{TA}	Y | RK_{TA}

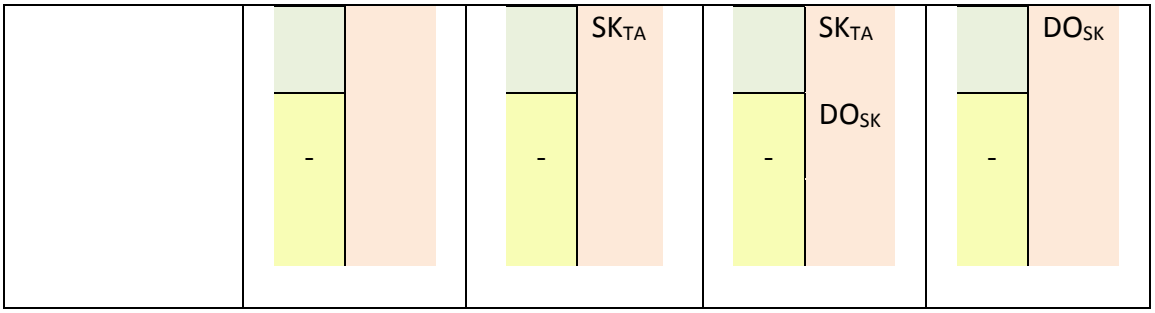

Table 14: Access Matrix (Embodiment 2)

It's important to note that each entity can access only a well-known state of the *data*:

- IP and TA, which act as semi-trusted entity, cannot access to any kind of *data*, but they can only perform some crypto functionalities on it.
- DC can access only DATA if it owns the rights to do it.
- MA, that act as an attacker, even if it can access to all of state of *data,* is not able to perform any operation that leads to obtaining of the original DATA.

7.8 DIFFERENCE FROM THE STATE OF THE ART

In Cloud computing there would be numerous infrastructure providers all providing services to millions of data consumers. It is simply not possible to re-encrypt data every-time a user has access revoked. As there would be many data consumers who would be having their access revoked in a very short span of time which means that there would be more than one re-encryption operation taking place. It is very hard if not impossible to keep track of which data is encrypted with which key. Therefore this approach presents a very practical and scalable solution to problem of hosting data on un-trusted infrastructure provider. Our scheme would scale relative to the state of the art schemes[96][97], as there are no lengthy complex re-encryption and key management operations that needs to be performed.

One of the biggest advantages of the scheme is that it requires significantly fewer key exchanges compared to the other schemes [171][172]. This feature is built into the scheme as it only requires that the data owner and trusted authority initially have a baseline level of trust so that their public keys can be shared. Afterwards each domain

that uses the trusted authority would provide the public keys of its data consumers itself. There are no requirements for distributing session or private keys.

Caching of frequently accessed files at the domain level would be used to ensure that less network level resources are used when a request comes in. If a request for the same file comes in from a different user all the domain administrator has to do is to forward the file to the user. It then notifies the data owner to release the key for the decryption of the file to the data consumer.

Furthermore, the scheme can be used in a setting where decryption is performed not at the data consumer level but at the domain level. For instance, Company A wants all the data to be re-encrypted using its private key, and when the Company A receives a file on behalf of a data consumer, it then performs decryption and forward it to the respective data consumer. The benefit of this approach would be that caching of files would not require provisioning of the keys by the data owner or decryption of the files, as if the request for the same file comes in, then all the domain administrator has to do is to forward that file to the appropriate data consumer without performing decryption.

7.9 PSEUDOCODE

What follows is the explanation of the processes of a new user joining and user revocation in the scheme ACDC^3.

New User Join: Every time a new data consumer wants to access files stored on the infrastructure provider it has to first request the administrator of the domain. The domain administrator then ensures that the trusted authority has access to appropriate credentials of the data consumer. The domain administrator provides a web based query service that provides appropriate credentials (Attributes and Public key relating to an identity) of the data consumer to the trusted authority. Trusted authority uses this service to check the credentials of data owners who want to access files. This service can be an LDAP server or an active directory server. Following is the Pseudo code of the new user join operation,

```
//Following function is called by the data consumer to initiate the process of user join

NewUserJoin(Name, EmployeeNumber)

{

    If (Name is in LDAPDirectory() and EmployeeNumber is in the
    EmployeeDirectory())

    Then

    {

        getAttributes (Name, EmployeeNumber)

        getPublicKey(Name, EmployeeNumber)

        UpdateDirectoryService(Attributes,PublicKey)

        //Updating Directory service that the trusted authority queries

    }

    Else {(Return (Wrong Name or Wrong Employee Number) }

}


//Following function is called by the trusted authority

/* Name represents the name of the entity that is calling the function like domain
administrator, data owner or trusted authority. Authentication is the process by which
the entity authenticates itself to the directory service and DCName is the name of the
data consumer to which the query is about.

*/

DirectoryService(TAName,Authentication,DCName)
```

```
{
        If(Authentication Fails)
        Then {Return (Authentication Failed)}
        Else
        {
                        If(DCName is in LDAPDirectory)
                        Return (Attributes,PublicKey)
                        Else {Return (WrongDCName)}
        }
}
```

User Revocation: The process of user revocation is initiated by the administrator of the domain. It notifies the trusted authority that the data consumer has no longer rights to access files on the infrastructure provider. The trusted authority then deletes the attributes and public key of the data consumer from its records.

The domain administrator also ensures that web based directory service no longer holds the credentials of the data consumer. Once these operations are complete then the data consumer access is revoked and he no longer can decrypt files stored at the infrastructure provider. Following is the pseudo code for the user revocation process,

```
/* Following function is called by the domain administrator to delete credentials from the web based directory service. */
DirectoryService(Name, Authentication,DCName)
{
```

```
        If(Authentication Fails)
        {Return (Authentication Failed)}
        Else
        {
        //Following function deletes credentials of the data consumer from the directory
        deleteCredential(DCName)
        }
}
```

/* Following function is called by the trusted authority to delete the data consumer attributes and public key from its records. */

```
UpdateRecords (DCName, Delete)
{
        If (DCName is in Direcotory.Name() )
        Then
        {
                deleteAttributes (DCName)
                deletePublicKey(DCName)
        }
        Else { Return (Incorrect DCName) }
}
```

7.10 FINE GRAINED ACCESS CONTROL

State of the art schemes use attributes to perform fine grained access control. These schemes achieve fine granularity by encrypting files using a keys that have attributes embedded in them. Only the data consumers who have the correct key with the appropriate attributes are able to decrypt the files [171] [97].

This approach is cumbersome and it requires user specific encryption to be performed per file. In our scheme we have delinked the fine granularity of access control with data confidentiality. This approach has enabled the scheme to perform fine grained access control at the trusted authority level. The biggest benefit of the approach is that it is less complex (computational overhead, time).

In ACDC^3, a centralised access control mechanism is used in which a fine grained access control policy is defined with respect to a domain. This approach enables us to update the access control policy, without having to re-encrypt all the files. Every domain represents an enterprise or collaboration, this domain has specific requirements with regards to the access control, using our mechanism it can define rich access control policies.

The mechanism is based on eXtensible Access Control Mark-up Language (XACML) [175], which is an access control policy framework based on three aspects.

Firstly it offers a policy language that can be used to express control rules and conditions. Each policy constitutes multiple rules and policies itself can be combined into sets. It offers a mechanism that represents the governance framework of an organisation (domain).

Secondly it offers a protocol to represent the request and response. Real world access control request can be constructed using the protocol. These request than go to an XACML engine for evaluation and the result is then returned which is normally permit, deny or in-applicable.

The third feature that XACML offers is reference architecture that proposes software modules to be deployed to ensure efficient implementation of security policies. The modules include, Policy Decision Point (PDP) that evaluates policies against access request. Policy Enforcement Point (PEP) which is responsible for providing the access requests. Finally the Policy Information Point (PIP) that is queried by PDP and PEP to gather information about subjects and the objects.

The advantage of using XACML is manifold, it offers a standardised approach to authorisation by which many different domains can be integrated without a lot of hassle and the focus is on the security policies rather than technicalities of the environment. Furthermore, XACML follows an attribute and policy based approach which makes it fine grained.

ACDC^3 achieves fine grained access control using XACML, but usage of meta-files in this scheme has a major drawback. The meta-files are not encrypted and they can be potentially read by the infrastructure provider. The infrastructure provider can learn some information about which files are accessed but it cannot learn anything about the encrypted files themselves. Furthermore, the kind of information that is revealed also depends upon the scenario and on the data consumer. A potential solution to this problem can be use of abbreviation rather than text in the meta-files. It would limit the learning capacity of the infrastructure provider. An implication of this approach is that the access control mechanism has to know which abbreviation means what in advance in order to interpret them.

7.11 SECURITY ANALYSIS

User Revocation: The benefit of ACDC^3 scheme is that user revocation is independent of data re-encryption by using proxy re-encryption to perform on the fly re-encryption. This reduces the computational overhead and simplifies the process of user revocation. The process of user revocation ensures that the data consumer who has it's access revoked cannot decrypt any information hosted on the infrastructure provider even if both of

them collude. Furthermore, the scheme ensures forward and backward secrecy even when a large number of revoked data consumers and infrastructure providers collude.

Single Point of Failure: The trusted authority represents a single point of failure for the whole scheme. In case the trusted authority goes down the whole scheme would no longer function. A potential solution to the problem is that the data owner ensures that backup of the trusted authority is made so that in case the data relating to security policies and keys on trusted authority is lost, it can be recovered. Furthermore, the data owner should also ensure that back up servers come online in case the main server is not working.

Fine-grained Access Control: ACDC^3 enables a data owner to deploy fine grained access control policies which are independent of data confidentiality. This ensures that rich policies are developed with focus on corporate governance rather than on the technicalities of cryptography and software. This setting is very suitable to the Cloud computing scenarios as there would be many enterprises (domains) that would be using the Cloud while acting as both data owner and data consumer.

Data Consumer Access Privilege Confidentiality: The major drawback of our scheme that we have already mentioned is the use of meta-files. However if we compare our scheme with other scheme (that use ABE) such as that of Yu et al [95], only the leaf nodes of the access tree are disclosed to the infrastructure providers in this scheme. Therefore, this scheme also reveals attribute information relating to data consumers to the infrastructure provider. Our scheme achieves similar levels of access privilege confidentiality as that of the previous schemes. However, it offers a less complex and richer mechanism for fine grained access control.

CHAPTER 8

EXPERIMENTAL VALIDATION

In this Chapter we present the experimental results of the ACDC[3] scheme and security risk framework. The Risk Assessment Framework implementations were validated by performing experiments within the OPTIMIS toolkit. The toolkit adopt use cases such as multi Cloud and enterprise Cloud broker as explained in Chapter 2.

8.1 RESEARCH GAPS

There were five research gaps that were identified as part of this thesis. In this section, the research gaps are revisited to link up the experimental validation chapter with the research gaps identified earlier.

RG1 is the gap relating to the confidentiality of data hosted on Cloud platforms. Cloud platforms where data is hosted are outside the physical control of the data owner. Therefore, ensuring confidentiality of the data is very important. Conventional mechanisms available for encryption can be applied to Cloud platforms but they cannot scale as they are not developed to handle huge amount of subscribers. Therefore, the requirement is to develop a scalable confidentiality scheme that can cater for the Cloud

computing scenario. Moreover, providing fine-grained access control and integrity of data is also part of this research gap.

Research gap RG4 highlights the issue of user revocation in Cloud computing scenario. When a user gets their access revoked, in conventional schemes, new keys are distributed to existing subscribers. The broadcast server then uses the new key to encrypt. This task is done to ensure that the revoked user can no longer access the system. The problem with this approach is that it is not scalable as a large number of users leaving the system regularly would require the scheme to follow the cumbersome process of generation, revocation and issuance of new keys.

To address these two research gaps (RG1, RG4) the scheme of ACDC^3 was developed. The scheme provides a mechanism to ensure confidentiality of data hosted on Cloud platforms whilst also providing user revocation. The results of the experiments with the scheme are presented in this Chapter (ACDC^3 Scheme).

Research gap RG5 highlights the problem of mitigating risk associated with Cloud computing platforms. An organisation moving to the Cloud needs to understand what sorts of risk they are taking. The Cloud-focussed risk framework presented in this thesis attempts to address this challenge.

To address research gap RG5, a framework was developed for risk identification, threat assessment and mitigation strategies for Cloud computing scenarios. For this, a risk analysis was carried out for four different Cloud settings. In this Chapter we present the results of the risk analysis performed using the IRAM toolkit. The results identify the categories of risk and their corresponding impact in the form of very high, high, medium and low. The prioritisation of identified risks is also performed.

To fulfil the research gap RG1 and RG4, the ACDC^3 scheme was developed. We present in this Chapter the results of evaluating the ACDC^3 scheme via a set of experiments. For the ACDC^3 scheme we have developed an experimentation model in which we draw two comparisons of the scheme. The first comparison is with symmetric encryption scheme AES and the second is when there is no encryption applied.

Cloud computing should create the illusion that the configuration relating to encryption, decryption, management of security policies and management of keys is done seamlessly. This illusion forms the basis of Cloud computing and it differentiates Cloud computing from other forms of distributed computing. The research gaps RG2 is to address this challenge by coming up with tools and technologies that would enable this automatic configuration. RG2 is not tackled as part of this thesis and remains part of the future work.

None of the analysed systems proposes a mechanism by which access control policies that are distributed over multiple infrastructure providers are kept synchronised. In Cloud computing scenarios, the data may reside and pass from numerous platforms like Broker, Service provider and Infrastructure provider (IP). The challenge is to ensure synchronisation of access control policies across these multiple domains. This research gap RG3 is also part of future work and is not tackled in this thesis.

8.2 ORIGINAL CONTRIBUTION

The Security Risk Framework was developed in collaboration with University of Leeds as part of the OPTIMIS project. The main contribution of the researcher came in the form of threat assessment, prioritisation of threats, designing the Security Risk Framework and threat inventory development. The implementation and algorithm development was done as joint work between the researcher and Mariam Kiran, the implementation was primarily led by Mariam whereas the researcher was in a support role.

For the $ACDC^3$ scheme, from the conception, design all the way to the implementation and experimentation is the sole work of the researcher. Franchesco La Torree did contribute in verifying the design of the scheme, but these results are not presented in this thesis.

8.3 SECURITY RISK FRAMEWORK

The Security Risk Framework described in Chapter 5 was developed as part of the Risk Assessment Framework of the OPTIMIS project. This section is presenting results of the threat assessment work undertaken in Chapter 5. The threat assessment was done to identify the threats relating to Cloud computing scenarios. This work was done using ISF IRAM toolkit. This section also presents the prioritisation of the Challenges that were identified as a result of the threat assessment and vulnerability assessment. This section has two main parts where the results are put forward. Sections are as follows,

- Threat Assessment
- Prioritisation of Challenges

8.3.1 THREAT ASSESSMENT

The risk assessment framework first requires that the main threats to Cloud computing scenarios be identified. For this the Information Security Forum IRAM tool was used. This tool comes with standard set of threats along with data relating to historical trends.

To undertake the threat assessment, following steps were taken

- Identification of threats relating to the Cloud computing scenario. For this a detailed threat assessment was performed for Cloud computing scenarios. As part of the threat assessment, the Cloud specific threats were prioritised
- Following the threat assessment vulnerability assessment was also provided for the Cloud use cases

In figure 30, we present the result of the analysis. Three threats have fallen in the category of very high; these are ‘Malfunction of system software’, ‘Malfunction of computer/ network equipment’ and ‘Gaining unauthorised access to systems or network’. Six threats are classified as high which are ‘Introducing malicious code’, ‘Distributing SPAM’, ‘Data leakage’, ‘Usage Control’, ‘Hypervisor level attacks’ and ‘User errors’. The rest of the threats fall in the categories of either medium or low.

These threats come directly from the threat assessment performed in Chapter 5 section 5.4.

Threat categories	A	B	C	D	E
External attack		2	8	2	
Theft			1	1	
System malfunction	2				
Service interruption			1	1	
Human error		1		1	
System-specific threat types		3			

Information risk ratings:
A - Very high, B - High, C - Medium, D - Low, E - Very low

Figure 30: Results of Threat Analysis

The identification and the categorisation of the above mentioned threats was instrumental in undertaking the risk assessment and also developing the security risk framework. This enabled the researcher to understand that main threats and vulnerabilities associated with Security Risk Framework.

8.3.2 PRIORITISATION OF CHALLENGES

From the threat analysis performed in the Chapter 5, we have shown that the information security principles of integrity, confidentiality and availability are most relevant to the Cloud related scenarios. The information risk ratings assigned shows that loss of confidentiality is rated as the highest level of risk followed by availability and integrity.

Using this analysis, the challenge of 'Access control in Cloud Computing' is linked with both availability and confidentiality. For instance, failure to ensure access control would mean the loss of availability in case a legitimate user is denied access, whereas unauthorised access would lead to the loss of confidentiality. Furthermore, an unauthorised user who has write access can update the data as well, therefore it could lead the loss of integrity. We have rated the challenge of 'Access control in Cloud

Computing' as the most important one as it has a direct impact on all of the security principles.

For the challenge 'Data Leakage Prevention' confidentiality is the main concern. An attacker who is able to leak confidential data by moving it and disclosing it to an unauthorised source would mean the loss of confidentiality. We have rated this challenge as the second most important as millions of users' data would be hosted on infrastructure providers. Ensuring that data is not leaked (especially of commercial nature) is of utmost importance. If this challenge is not addressed customers would lose confidence in infrastructure providers and that could potentially make them move away from hosting data on the Cloud.

'Hypervisor based IDS' and 'Hypervisor level Security' impacts confidentiality and integrity of data. We have rated 'Hypervisor based IDS' as the third most important challenge as it would be highly desirable to detect and delete worms and viruses from hypervisor level. This functionality would act as an added on functionality to supplement other security functionalities. 'Hypervisor level Security' is an added functionality; not providing it may not have a similar impact as the first two challenges have.

The challenge ratings are summarised as follows.

1. Access control in Cloud Computing (Confidentiality, Availability, Integrity)
2. Data Leakage Prevention (Confidentiality, Integrity)
3. Hypervisor based IDS (Confidentiality, Integrity)
4. Hypervisor level Security (Confidentiality, Integrity)

8.4 ACDC^3 SCHEME

The ACDC^3 scheme design, architecture and the research problem that it is solving is detailed in Chapter 7. In this section we provide the result relating to the experimentation and implementation of the scheme using the NICS Crypto Library[176]. For the

experimentation the focus of this section remains on the confidentiality, user revocation and efficiency perspectives. The experimentation conducted for the ACDC^3 scheme are designed to compare the scenarios where there is no encryption and there is encryption using symmetric keys.

8.4.1 NICS CRYPTO LIBRARY

NICS Crypto Library was developed primarily for an OpenID solution. The main characteristic of the solution was that it was privacy preserving. The solution enabled an Identity Provider to give attributes to other parties without being able to read their values[177].

The NICS Crypto Library has three main programming modules, which are as follows,

- Global Parameters
- Proxy Re-Encryption
- Main Java File

The Global Parameters module assumes that all the global parameters are known by all the parties involved in the scenario. Effectively, the library hard-codes the parameters such that it can present the results required for the OpenID solution. For the Proxy Re-Encryption module an implementation of the proxy encryption scheme is done. The main Java file is used to run the scheme and provide it with initialisation vectors.

As explained above the library was designed for the OpenID solution therefore, we had to write our own Java module for it to work.

8.4.2 ACDC^3 JAVA MODULE

The code developed for the ACDC3 follows the steps mentioned below,

1. The program starts by creating global parameters that will be used to create the keys. The global parameters are then passed to the initialisation module of the library to start the setup stage.

2. The second step is to generate keys relating to the data owner, data consumer, trusted authority and infrastructure provider.
3. As the scheme is based on Ateniese Scheme (Embodiment 2) it requires trusted authority private key and data owner's public key for this for the generation of re-encryption keys. Once this information is provided Re-encryption keys are generated.
4. Encrypt the message m using the core encryption key.
5. Encrypt m with Data Owner's public key, at the trusted authority level.
6. Re-encrypt the tuple with the re-Encryption key at the trusted authority, result will be another Cipher text. This is the process of re-encryption through which one cipher text transforms into another.
7. Send the Cipher text to infrastructure provider. This we simulate the serializing of the cipher and storing it in a file.
8. Once the request for access comes in from data consumer, download the cipher from the infrastructure provider.
9. Now using the trusted authority re-encryption key, transform the cipher text from CT'' to CT'.
10. In the final step, pass the CT' to the data consumer.
11. Data consumer at this stage will decrypt the file using the core decryption key.

Refer to Chapter 7 for a detailed description of the scheme and its design. The code of the scheme is present in the annex section of the thesis. For the implementation there were numerous issues. For instance, since the NICS crypto library was used the implementation scope was stuck with what the library has to offer. Moreover, we were also bound to write a single Java application as the library was primarily used to be run on single server. This limited our capacity to run experiments on different Cloud platforms. The implementation is a simulation as we have all the players of the scenario residing on the same platform.

8.4.3 DESIGNING THE EXPERIMENTS

The purpose of the experiments is to demonstrate the efficiency of the ACDC^3 scheme compared against standard encryption and no encryption. The experiments are designed in such a way that the actor Data Owner, Data Consumer, Infrastructure Provider and Trusted Authority reside on the same computer. For each of the actor time that is recorded for uploading and downloading of the file. The detail of the scheme itself and its mathematical formalisation is present in Chapter 7.

To calculate the efficiency of the scheme the comparison is done between two scenarios. One scenario is that when there is no encryption, the other scenario is that when there is symmetric encryption. Results are compiled against these two scenarios.

The experiment run as a simulation, the characteristics of the machine are as follows,

- Processor: Intel Core i5 1.7 GHZ
- RAM: 8GB
- System Type: 64 bit Operating System Windows Enterprise 8.1

The testing strategy of the scheme was divided into three parts,

- The first experiment is to do with the uploading and downloading of a file on the Cloud. There is no encryption involved in this stage, therefore only the upload and download times are recorded. Finally the total time of upload and download is also calculated.
- The second experiment starts by encrypting a file through normal symmetric encryption using 128 Bit Key. The algorithm used is AES, padding is PKCS5 and the mode of encryption is CBC. CBC mode of encryption is used because its adds randomness to the cipher text. The ECB mode of encryption always creates the same cipher for the same plain text as it does not add padding. This makes the encryption weak. Afterwards the file is uploaded on the Cloud. The upload takes place from Data owner to Infrastructure provider. The third step is when a request for accessing the file comes from Data Consumer. The file is downloaded from

Infrastructure provider to Data consumer. In the last step the file is decrypted by the Data consumer.

- The third experiment encompass the processes of the ACDC^3 scheme as explained in the previous section. These steps would include initialisation, key generation, re-encryption and decryption.

The three above test cases were developed to find out the performance overhead in terms of encryption and decryption for the ACDC^3 scheme.

The data size that is selected for encryption is 200 bytes, 400 bytes, 800 bytes, 1600 bytes and 3200 bytes. The reason we went along with these sizes is because comparatively the size difference between them is enough to show us the correlation that exist. Rather than taking different sizes our focus was on running the experiment on the same sizes ten times. This would enable us to gather the average time taken for each size. Moreover, we have limitation from a resource perspective that would require numerous testbed machines hosted on different platforms. Furthermore, from an implementation perspective we have the limitation that increasing the size of the files was crashing the program. The program require recoding and it would have to be deployed on a more powerful system for it to work. The variance that existed when running the experiment on the same size was significant, therefore it was imperative to run the test many times. Figure 33 shows the output of the code as per the experiments,

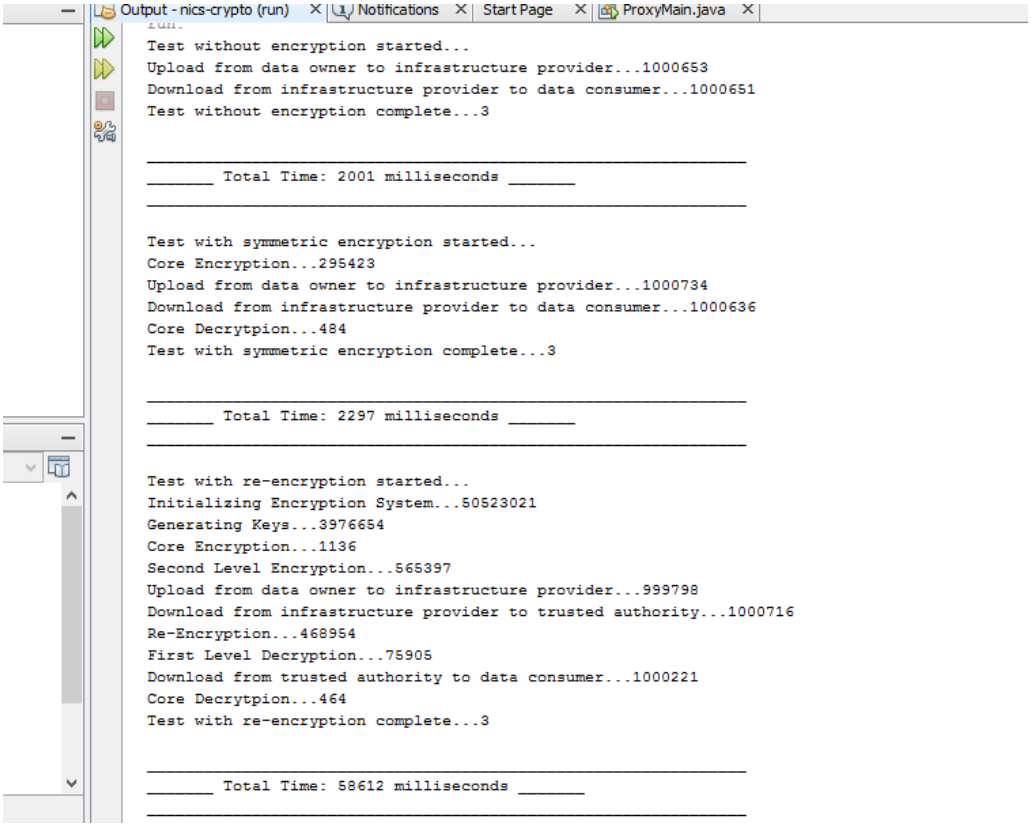

Figure 31: Output of the Experimentation

Figure 31 presents the time taken by different stages of the simulator. For instance time taken to perform core encryption, upload the file to the server, download the file from the server, performing decryption. These stages are for symmetric encryption whereas other stages are added for the ACDC3 scheme.

8.4.4 Testing

The results for the experimentation done is divided as the different file sizes, the tables 15 and 16 show the results of the experiment for 3.2 KB size only. Similar test were conducted for the sizes of 1.6KB. 800 Bytes, 400 Bytes and 200 Bytes. A total of 50 experiments were conducted in order to determine the average time for encryption, decryption and other stages of the ACDC3 scheme with respect to different data size.

The times are calculated in nanoseconds for all the values apart from total values which are recorded in milliseconds. The experiments are conducted for three scenarios,

“Without Encryption”, “With Symmetric Encryption” and “With Re-Encryption Embodiment 2”. The table 15 comprise of the scenarios of “With Encryption” and “With Symmetric Encryption” whereas the table 16 comprise of the scenario of “With Re-Encryption Embodiment 2”.

Parameter	Without Encryption			With Symmetric Encryption			
Length of Bit Stream	*Upload DO to IP*	*Download IP to DC*	*Total*	*Encryption*	*Upload DO to IP*	*Download IP to DC*	*Decryption*
(3.2 KB)	988714	1000143	1988	268968	996096	1000169	641
(3.2 KB)	1000266	1000403	2000	271560	999752	1005947	662
(3.2KB)	1000611	1009114	2009	270957	1006127	1000037	627
(3.2 KB)	1000375	1000152	2000	269811	1000068	1000169	614
(3.2 KB)	1013565	1000186	2013	672847	999802	1006873	639
(3.2 KB)	1008885	1000083	2008	901338	1015286	1000050	613
(3.2 KB)	1000662	1000347	2001	273113	1004803	1003405	697
(3.2 KB)	1009739	1000027	2009	255492	994082	1008877	687
(3.2 KB)	1008587	999996	2008	267376	1010097	1000078	716
(3.2 KB)	1003973	1000137	2004	269336	1013381	1000131	706
Total	**10035377**	**10010588**	**20040**	**3720798**	**10039494**	**10025736**	**6602**

Table 15: 3.2 KB size, showing time for Without Encryption and with Symmetric Encryption

With Re-Encryption (Embodiment 2)										
Initialization	*Generating Keys*	*Core Encryption*	*Second Level Encryption*	*Upload DO to IP*	*Download IP to TA*	*Re-Encryption*	*First Level Decryption*	*Download from TA to DC*	*Core Decryption*	*Total*
189951619	3953162	1419	786489	1001296	1000242	469816	78688	1005894	838	198249
14863825	4052642	1402	458594	999716	1009373	480926	75896	1010495	915	22953
68016952	3980669	1302	438492	1008125	1000239	483012	76417	1013441	875	76019
256102266	3999599	1418	435499	1000140	1000176	480996	77526	1000013	844	264098
86304549	3280095	1329	437394	1000170	1007341	479224	80372	1004721	907	93596
71937456	3269956	1336	820095	1009001	1000036	637350	235747	999931	906	79911
19640930	3954152	1342	433616	1000037	1000479	476855	78820	1001139	926	27588
16857050	3926251	1355	435151	1014899	999968	474542	77822	1012689	869	24800
91024399	3923999	1342	421653	1012737	1000466	464973	74296	999734	644	98924
101025726	3951160	1445	493778	1000117	1010705	830361	234132	999870	992	109548
915724772	**38291685**	**13690**	**5160761**	**10046238**	**10029025**	**5278055**	**1089716**	**10047927**	**8716**	**995686**

Table 16: 3.2 KB size, showing time for ACDC3 scheme

Similar test were run for values of 1.6KB, 800 Bytes, 400 Bytes and 200 Bytes. After running the test, average time was calculated for each size which is then stored in the table 16.

There is variance in the time values of the different stages of the same operation. This is due to the processor of the server performing other processing tasks at the time it is undertaking encryption, decryption operations etc.

8.4.5 Results of the Experiments

For all the tests that were run, average time was calculated for encryption for each size against the three different experiments mentioned in the previous section.

X-axis represents the size of the file in kilobytes whereas the Y-axis presents the time in milliseconds. The results of the experiments are as follows:

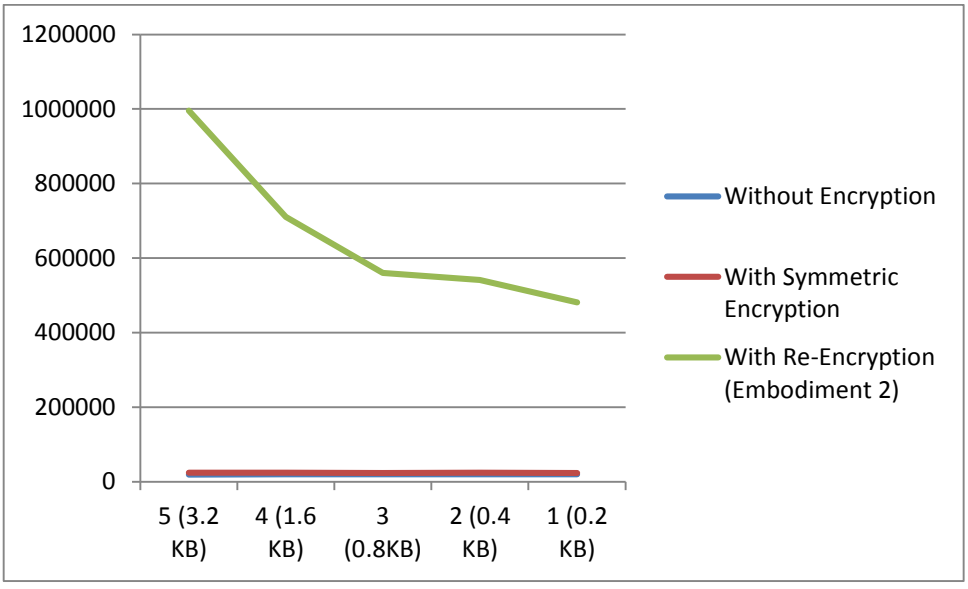

Figure 32: Graph for different times recorded for the three scenarios. X-axis showing data size in kilobytes, Y-axis showing time in milliseconds for encryption

The figure 32 show that the time recorded for symmetric encryption and for no encryption is very similar. They both fall on the same line; this is because symmetric encryption is highly efficient. The time for ACDC[3] is changing with the size of the data as

shown above. It is pertinent to mention here that we do not include the complexities that come with the usage of symmetric encryption. The deployment of PKI infrastructure and distribution of keys. However for the ACDC^3 these complexities are accounted for. Moreover, the encryption process for the ACDC^3 scheme is a one off operation.

Figure 33 below shows the total time taken for re-encryption stage in the ACDC^3 scheme against data size. The graph shows that the size of the data doesn't have much impact, as the re-encryption operation that is performed is pretty efficient. It depends more on the processor time and the number of other jobs running when this experiment is conducted. In this experiment the X-axis presents the size of the file in kilobytes whereas the Y-axis presents the time in milliseconds.

The results presented in the figure 33 are very important from the perspective that the size of the data has little or no impact on the re-encryption operations. Therefore, this experiment validates that the re-encryption process is agnostic to the data size. Also validates the claims with regards to the efficiency and scalability of the scheme. The re-encryption process would be required for all the data that is being encrypted. Therefore, it will be repeated every time file is accessed.

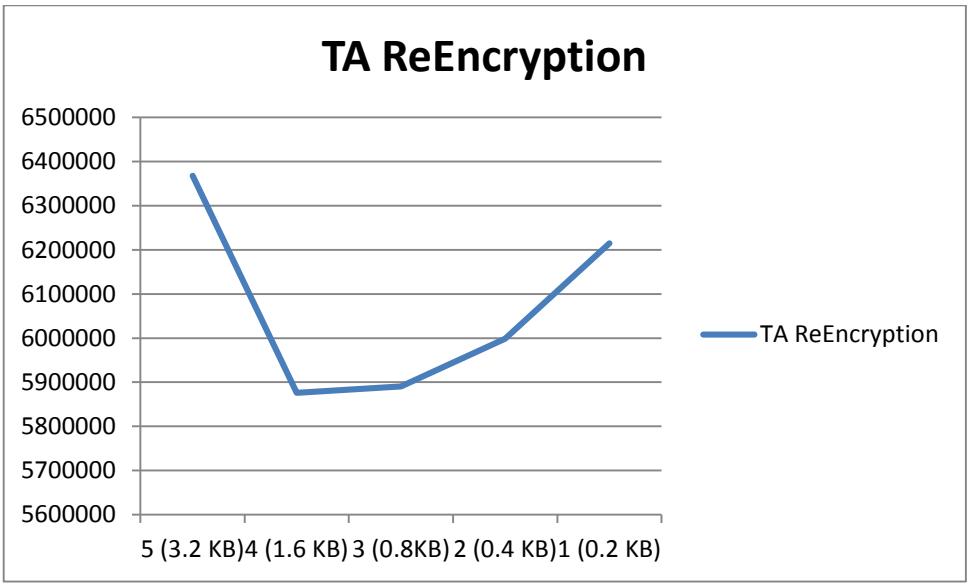

Figure 33: Graph of data size against Re-Encryption time. X-axis showing data size in kilobytes, Y-axis showing time in milliseconds for re-encryption

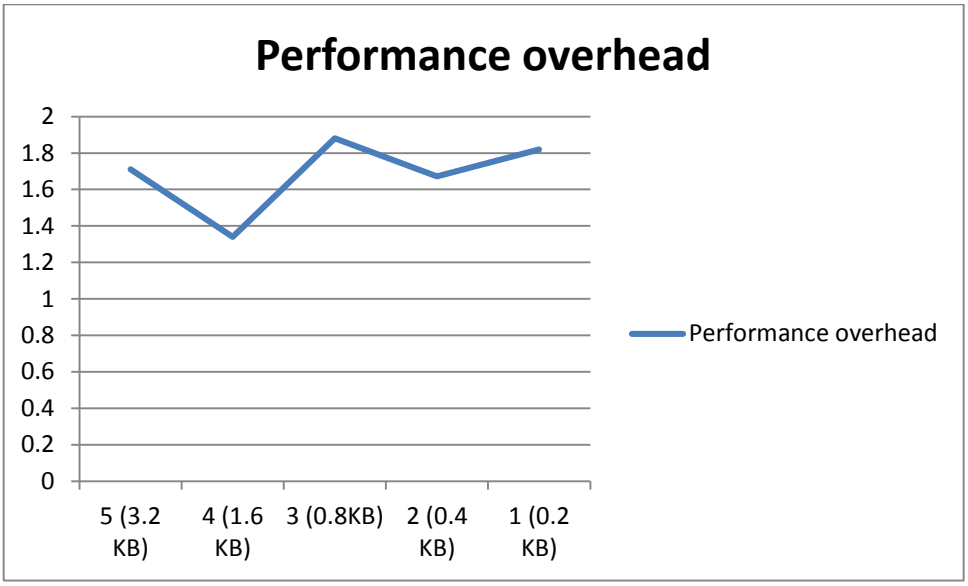

Figure 34: Comparing the decryption times against data size X-axis showing data size in kilobytes, Y-axis showing time in milliseconds for the performance overhead

As in the ACDC[3] scheme, the encryption is a one off process, it is undertaken by the data owner at the start of the scheme. Therefore, the actual performance overhead is calculated by comparing the encryption plus decryption (E+D) time of symmetric encryption against ACDC[3] scheme decryption steps. The decryption steps include re-encryption, first level decryption and core decryption. Figure 34 shows the performance overhead of the ACDC[3] scheme when compare against symmetric encryption. The figure 34 results show that the performance overhead is on average 1.6 times compare with symmetric encryption. In this analysis we are not catering for the time that symmetric encryption would take when there is going to be a user revocation.

As explained before, when a user revocation is done, data re-encryption is required in the symmetric encryption setup. As the user leaving the trust domain still have the key that it can use to decrypt data. Moreover, the process of re-encrypting the data gets more complicated as the number of users go up in the symmetric encryption setup making the symmetric encryption non-scalable.

In the ACDC[3] scheme setup, there is no such complication relating to user revocation, the data is encrypted once and no re-encryption is required when a user revocation takes

place. Therefore, arguably ACDC^3 would scale in a Cloud computing environment compared with symmetric or asymmetric encryption. The decryption process would be required whenever there is a file access and it is not a one off process.

8.5 CONCLUSION

As part of this research work the Security Risk Framework was implemented and integrated in the OPTIMIS project. Moreover, the threat analysis performed enable the identification of risk and their prioritisation as per the OPTIMIS use cases. The Security Risk Assessment model was implemented until the deployment phase only and presented as part of the risk assessor at deployment stage. At operation stage its implementation work is planned for future research issues, with monitoring infrastructure providing real-time data for the risk assessor for continuous security risk assessment.

As for the ACDC^3 scheme, the implementation of the scheme was done using the NICS Crypto library. After the implementation experiments were designed for three scenarios, No Encryption, Encryption using Symmetric Cryptography and ACDC^3 scheme. Results show that ACDC^3 is 1.6 times slower than the symmetric encryption in a setting where we are not taking into account key management complexity relating to the symmetric cryptography in a Cloud environment. The ACDC^3 does however, resolve the issue of user revocation and provides a scalable solution in the Cloud computing environment.

CHAPTER 9

CONCLUSIONS

9.1 SUMMARY

This thesis has presented research that takes forward the engineering of secure Cloud systems. Our review work identified a variety of "research gaps" that drove the direction of the actual research conducted. Three specific research contributions were made (detailed in chapters 5, 6, and 7). Below we restate what the research gaps were, summarise the success of the work carried out, summarise limitations of the work and its evaluation, and provide pointers for future work.

- The aim of the thesis was to undertake research in the area of Cloud computing security. From the start of the project, it had two tiers: an industrial one and an academic one. The industrial one was focusing on the known challenges that the industry was facing whereas from the academic point of view the whole idea of Cloud computing was questioned.
- This thesis starts with highlighting Cloud definitions, differentiation between Cloud computing and other forms of computing. From an industrial perspective

the evolution of Cloud computing was explained. This is followed by a state of the art implementation review of Cloud computing delivery models and their security specification. The idea was to develop a solid understanding of the Cloud computing products and their limitations.

- The next logical step was to identify the state of the art from the academic perspective. This would mean identifying research gaps. For this, a thorough literature review was undertaken where the research gaps of scalability, user revocation and security risk management for cloud computing scenarios were identified.
- The research gap of risk management is filled by developing a Security Risk Framework, which allows monitoring of threats based on the events being logged by the detectors leading to a calculation of the relative risk. The relative risk is a barometer that enables a computer administrator to mitigate the risk by taking corrective actions or accepting the risk. For the Security Risk Framework a threat analysis is performed that leads to the identification of major threats relating to Cloud computing. The threats are prioritised and used in subsequent parts of the framework.
- The thesis also presents how one can secure a video broadcast subscription service in the Cloud computing setup. The scalable video scenario is built on top of an IaaS Cloud and shown how a video can be encrypted and authenticated efficiently.
- To solve the challenge of scalability, user revocation and data confidentiality in Cloud computing. We have developed a novel scheme ACDC^3, to the best of our knowledge it is the only scheme that achieves efficient user revocation, data confidentiality and scalability simultaneously for access control scenarios in Cloud computing. We have achieved this objective by using Ateniese proxy re-encryption (PRE) and by decoupling the operations of encryption and access control.
- In the final part of this thesis, the experimental validation for the Security Risk Framework and ACDC^3 scheme is detailed. The Security Risk Framework was

implemented and integrated in the OPTIMIS project. Moreover, the threat analysis performed enable the identification of risk and their prioritisation as per the OPTIMIS use cases. As for the ACDC^3 scheme, the testing showed there is a performance overhead of 1.6 times on average in absolute terms. This is in comparison with symmetric cryptography. However, the complexity of key management and data re-encryption within the symmetric cryptography makes it unviable for Cloud platforms due to scalability issue. This problem of scalability and user revocation are resolved by the ACDC^3 scheme. The ACDC^3 scheme would fare well in scenarios where the number of subscribers are in the millions and there is frequent requirement for joining and leaving of the subscribers. The scenarios where the subscribers are relatively small and the joining and leaving operations are not frequent the symmetric encryption scheme would be more efficient.

- The research undertaken in the EngD was published in four research papers. From an industrial perspective two patents were developed from the research work. Furthermore, the research work made direct contributions towards the Cloud strategy of BT through numerous papers and deliverables.

9.2 LIMITATIONS

For the security risk framework the experiments were conducted using the OPTIMIS testbed. The OPTIMIS project came to end two years ago. There is a major limitation as far as testing is concerned for the security risk framework as there is no testbed where further testing can be done.

The testing of the security risk framework was confined to the deployment stage and not to the operational stage. The testing during the operational stage would have potentially highlighted further issues with the scheme.

For ACDC^3 a major limitation was that of testing the scheme on numerous cloud platforms while it is providing a service widely consumed. The testing was carried out via

a simulation on a single machine of users accessing services via the scheme. In practice, of course, the scheme would be implemented across platforms and clients and various server agents would not be co-located. A more realistic deployment would likely raise practical integration issues associated with the scheme. The testing was done on data size which ranges from 200 bytes to 3200 bytes. The program was not able to handle larger size of data sets as it crashed. This was due to hardware limitation and also the implementation requires re-coding. We do believe that larger data sets should be tested to better understand the robustness of the scheme.

The fine granularity part of the ACDC^3 scheme was not developed. Therefore it would be required to take an already existing implementation of XACML and integrate with ACDC^3 or to implement a new module within the ACDC^3 scheme. The testing was largely aimed at evaluating the confidentiality and availability features of the scheme and not the fine granularity ones.

User revocation for ACDC^3 scheme was not experimentally validated. For this validation it would be required to setup large number of users on Cloud infrastructure. These users would consume the cloud services using the communication protocols laid out in the ACDC^3 scheme. A program monitoring the scheme would have to be developed which can then record the metrics of efficiency and confidentiality. This program would also validate that the user revocation functionality of the scheme is working as per the mathematical formalisation mentioned in Chapter 7.

9.3 FUTURE WORK

The future work for Security Risk Framework includes testing the system on a Cloud platform with monitoring agents installed which will log certain threats when they occur. This will then be extended to work on determine threats which may be eventually seen based on the data being collected and difficult to determine directly from the events. Finally the results from the testing and evaluation, advocate that the risk model does correctly assess and prioritize the risk.

The Security Risk Assessment model implementation needs to be extended to encompass the Operational Phase, where run-time monitoring agents could be deployed to enable continuous security risk assessment. It is envisioned that the security risk framework should be an open source code that could be integrated with other Cloud adoption toolkits. It would enable the decision makers to make sound decisions on Cloud adoption from an information security perspective.

For the Scalable video scenario future work would be focused on calculating the time delay and efficiency affecting the setup of the video broadcasting and predicting how this will affect the performance of the distributions.

The future work for the $ACDC^3$ scheme is around the development of prototype that sits on different Cloud platforms (Amazon, Google) and ensures user revocation and scalability while ensuring confidentiality of data. It would be desirable for the future work to expand the experiments that we have done for the $ACDC^3$ scheme by enhancing the size of the files that were used and also by having a larger number of simultaneous users of the scheme. Moreover, the implementation and integration of the fine granular part of the scheme is required. This would enable the testing of not only the scalability part of the scheme but also the fine granular part as well. Once this implementation and integration is complete the scheme would be in a suitable state to release commercially.

APPENDICES

SECURITY RISK FRAMEWORK CODE AND LOGS

This sections details the code written for the Security Risk Framework. Furthermore the logs section show a sample experimentation for the security risk framework.

RISK MODEL FOR THE SECURITY RISK FRAMEWORK

```
package securityriskjavamodel;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.Random;

public class RiskModel {


   public RiskModel() {
      System.out.println("hello from security risk model");
      // EvaluationMatrix matrix=new EvaluationMatrix();

   }



   public double getsecurity_risk(int ausecase, List threatList) throws Exception //returns the
total number of threats identified
   {

      int t_count = 0;
```

```
//connect to database and get the list of threats

//Iterator collectedthreatListIterator = threatList.iterator();

System.out.println("Total number of threats found"+threatList.size());

//parse list

double prob_likelihood, prob_priority;
double prob_threat;
double sum_prob_threat = 0.0;
double prob_inverse;
double prob_b;

int i;
//EvaluationMatrix matrix=new EvaluationMatrix();
for (i = 0; i < threatList.size(); i++) {
   ThreatData tdz=new ThreatData();
     tdz=(ThreatData)threatList.get(i);

     if (((tdz.get_usecase() == ausecase) || (tdz.get_usecase() == 0)) && ((tdz.get_stage() ==
0) || (tdz.get_stage() == 2))) {

//matrix.setEvaluationMatrixValue(ts,reference[i].get_likelihood(),reference[i].get_priority());
          prob_likelihood = (double) tdz.get_likelihood() / 5.0;
          prob_priority = (double) tdz.get_priority() / 5.0;
          prob_inverse = 1.0 - prob_priority;
```

```
            prob_b = (prob_likelihood * prob_priority) + prob_inverse;
            prob_threat = (prob_likelihood * prob_priority) / prob_b;
            sum_prob_threat += prob_threat;
            t_count++;
          }
        }



        //printout the matrix

        System.out.println("Total number of threats found " + t_count);
        sum_prob_threat = sum_prob_threat / (double) t_count;
        System.out.println("prod=" + sum_prob_threat);
        return sum_prob_threat;


    }

    // function 6

    public double getsecurity_risk_operation(int ausecase, double total_prob, List threatList)
throws Exception {
        int t_count = 0;
        //connect to database and get the list of threats
        int i;
        for (i = 0; i < threatList.size(); i++) {
           ThreatData tdz=new ThreatData();
             tdz=(ThreatData)threatList.get(i);
        }
        Random var = new Random();
        double number = 0.0;
```

```
//now traverse the list to form matrix

double prob_likelihood, prob_priority;
double prob_threat;
double sum_prob_threat = 0.0;
double prob_inverse;
double prob_b;
double relative_risk = 0.0, total_event_rate = 0.0;
//EvaluationMatrix matrix=new EvaluationMatrix();
for (i = 0; i < threatList.size(); i++) {
   ThreatData tdz=new ThreatData();
     tdz=(ThreatData)threatList.get(i);
   if (((tdz.get_usecase() == ausecase) || (tdz.get_usecase() == 0)) && ((tdz.get_stage() ==
1) || (tdz.get_stage() == 2))) {
      //Integer ti= reference[i].get_threat_id();
      //String ts=ti.toString();

//matrix.setEvaluationMatrixValue(ts,reference[i].get_likelihood(),reference[i].get_priority());
      prob_likelihood = (double) tdz.get_likelihood() / 5.0;
      prob_priority = (double) tdz.get_priority()/ 5.0;
      prob_inverse = 1.0 - prob_priority;
      prob_b = (prob_likelihood * prob_priority) + prob_inverse;
      prob_threat = (prob_likelihood * prob_priority) / prob_b;
      sum_prob_threat += prob_threat;
      t_count++;
      System.out.println("Total number of threats found threat: " + tdz.get_name());



      //check monitoring inputs
```

```
        }
      }



      System.out.println("Total number of threats found "+ t_count);
      sum_prob_threat = sum_prob_threat / (double) t_count;
      //System.out.println("Security risk monitored for usecase function 6  is " +
sum_prob_threat);
      // add monitoring loop
      int count = 0;
      for (i = 0; i < 10; i++) {
        number = var.nextDouble();
        if (number > 0.5) {
          count++;
        }
      }

      //System.out.println("Number of events recorded "+ count);
      total_event_rate = (double) count / 10.0; //assumption last 10 counts for service
      relative_risk = total_event_rate / sum_prob_threat;
      //System.out.println("Total event rate is in the last 10 counts "+ total_event_rate );
      //System.out.println("Relative risk calculated is "+ relative_risk );
      //System.out.println("If RR=1 do nothing, RR<1 accept risk, If RR>1 apply mitigation
strategy");
      return sum_prob_threat;


   }

}
```

JAVA SECURITY RISK PACKAGE

```
package securityriskjavamodel;


import java.io.File;
import java.util.ArrayList;
import java.util.List;
import org.w3c.dom.Document;
import org.w3c.dom.*;

import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import org.xml.sax.SAXException;
import org.xml.sax.SAXParseException;

import securityriskjavamodel.ThreatData;

public class SecurityRiskJavaModel {

   /**
    * @param args the command line arguments
    */
   public static void main(String[] args) {
      // TODO code application logic here

      //read the xml and create a list
         List<ThreatData> threatList=new ArrayList<ThreatData>();



      try
      {
```

```
DocumentBuilderFactory docBuilderFactory = DocumentBuilderFactory.newInstance();
DocumentBuilder docBuilder = docBuilderFactory.newDocumentBuilder();
Document doc = docBuilder.parse (new File("src/resources/threatdatabase.xml"));

 // normalize text representation
doc.getDocumentElement ().normalize ();
System.out.println ("Root element of the doc is " +
    doc.getDocumentElement().getNodeName());


NodeList listOfThreats = doc.getElementsByTagName("threat");
int totalThreats = listOfThreats.getLength();
System.out.println("Total no of threats : " + totalThreats);


int threat_id=0;
String name=" ";
int usecase=0 ;
int stage= 0;
String asset= " ";
int priority=0;
int likelihood=0;

for(int s=0; s<listOfThreats.getLength() ; s++){


   Node firstThreatNode = listOfThreats.item(s);
   if(firstThreatNode.getNodeType() == Node.ELEMENT_NODE){
      String t=null;

      Element firstThreatElement = (Element)firstThreatNode;

      //-------
```

```
            NodeList firstthreadIDList =
firstThreatElement.getElementsByTagName("threat_id");
            Element firstthreadIDElement = (Element)firstthreadIDList.item(0);

            NodeList textthreadIDList = firstthreadIDElement.getChildNodes();
            System.out.println("ThreatID : " +
                ((Node)textthreadIDList.item(0)).getNodeValue().trim());
            t=((Node)textthreadIDList.item(0)).getNodeValue().trim();
            threat_id=Integer.valueOf(t);

            //-------
            NodeList tnameList = firstThreatElement.getElementsByTagName("name");
            Element tnameElement = (Element)tnameList.item(0);

            NodeList texttnameList = tnameElement.getChildNodes();
            System.out.println("Last Name : " +
                ((Node)texttnameList.item(0)).getNodeValue().trim());
            name=((Node)texttnameList.item(0)).getNodeValue().trim();

            //----
            NodeList usecaseList = firstThreatElement.getElementsByTagName("usecase");
            Element usecaseElement = (Element)usecaseList.item(0);

            NodeList textusecaseList = usecaseElement.getChildNodes();
            System.out.println("Age : " +
                ((Node)textusecaseList.item(0)).getNodeValue().trim());
            t=((Node)textusecaseList.item(0)).getNodeValue().trim();
            usecase=Integer.valueOf(t);

            //------

             NodeList stageList = firstThreatElement.getElementsByTagName("stage");
            Element stageElement = (Element)stageList.item(0);
```

```
NodeList textstageList = stageElement.getChildNodes();
System.out.println("stage : " +
     ((Node)textstageList.item(0)).getNodeValue().trim());
t=((Node)textstageList.item(0)).getNodeValue().trim();
stage=Integer.valueOf(t);

//------

  NodeList assetList = firstThreatElement.getElementsByTagName("asset");
Element assetElement = (Element)assetList.item(0);

NodeList textassetList = assetElement.getChildNodes();
System.out.println("asset: " +
     ((Node)textassetList.item(0)).getNodeValue().trim());
asset=((Node)textassetList.item(0)).getNodeValue().trim();

 //------

  NodeList priorityList = firstThreatElement.getElementsByTagName("priority");
Element priorityElement = (Element)priorityList.item(0);

NodeList textpriorityList =priorityElement.getChildNodes();
System.out.println("priority : " +
     ((Node)textpriorityList.item(0)).getNodeValue().trim());
t=((Node)textpriorityList.item(0)).getNodeValue().trim();
priority=Integer.valueOf(t);

  //------

  NodeList likelihoodList = firstThreatElement.getElementsByTagName("likelihood");
Element likelihoodElement = (Element)likelihoodList.item(0);
```

```
            NodeList textlikelihoodList =likelihoodElement.getChildNodes();
            System.out.println("likelihood : " +
                ((Node)textlikelihoodList.item(0)).getNodeValue().trim());
            t=((Node)textlikelihoodList.item(0)).getNodeValue().trim();
            likelihood=Integer.valueOf(t);

            ThreatData ntd=new ThreatData(threat_id, name, usecase, stage, asset,
priority,likelihood);
            threatList.add(ntd);


          }//end of if clause


        }//end of for loop with s var

    }
    catch (SAXParseException err) {
    System.out.println ("** Parsing error" + ", line "
       + err.getLineNumber () + ", uri " + err.getSystemId ());
    System.out.println(" " + err.getMessage ());

    }
    catch (SAXException e) {
    Exception x = e.getException ();
    ((x == null) ? e : x).printStackTrace ();

    }
    catch (Throwable t) {
    t.printStackTrace ();
    }

    System.out.println("Size of threat list" +  threatList.size());
```

```
    RiskModel rm=new RiskModel();

    //calculate security risk at deployment
    double dep=0.0;
    int u= 2;
    try
    {
    dep= rm.getsecurity_risk(u, threatList);
     System.out.println("Security risk for usecase Private is " + dep);
    }
    catch(Exception e)
    {
    }



   try{
      double stagesix=rm.getsecurity_risk_operation(u,dep, threatList);
      System.out.println("Security risk for usecase function 6 is " + stagesix);
   }
   catch(Exception e)
   {
      System.out.println("print "+ e);
   }


  }


}
```

THREAT DATA JAVA CODE

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package securityriskjavamodel;

public class ThreatData {

   int threat_id=0;
      String name=" ";
      int usecase=0 ;
      int stage= 0;
      String asset= " ";
      int priority=0;
      int likelihood=0;

      public ThreatData()
      {
      }

   public ThreatData(int athreat_id, String aname, int ausecase, int astage, String aasset, int
apriority, int alikelihood)
   {
      threat_id=athreat_id;
      name=aname;
      usecase=ausecase;
      stage= astage;
      asset= aasset;
      priority=apriority;
      likelihood=alikelihood;
```

```
}

public int get_threat_id()
{
   return threat_id;
}

public String get_name()
{
   return name;
}


public int get_usecase()
{
return usecase;
}

public int get_stage()
{
   return stage;

}

public String get_asset()
{
   return asset;
}


public int get_priority()
{
   return priority;
```

```
}

public int get_likelihood()
{
   return likelihood;
}


 public void set_threat_id(int athreat_id)
{
   threat_id=athreat_id;
}

public void set_name(String aname)
{
   name=aname;
}


public void set_usecase(int ausecase)
{
   usecase=ausecase;
}

public void set_stage(int astage)
{
   stage=astage;

}

public void set_asset(String aasset)
{
   asset=aasset;
```

```
    }


    public void set_priority(int apriority)
    {
       priority=apriority;
    }

    public void set_likelihood(int alikelihood)
    {
       likelihood=alikelihood;
    }

}
```

LOGS FOR SECURITY RISK FRAMEWORK EXPERIMENTATION

Monitoring logs used for experimentation are as follows,

Extract from log:

```
104 01/23/1998 16:59:09 00:00:02 http 1879 80 192.168.1.30 192.168.0.40 0 -

105 01/23/1998 16:59:12 00:00:02 http 1880 80 192.168.1.30 192.168.0.40 0 -

107 01/23/1998 16:59:15 00:00:01 http 1881 80 192.168.1.30 192.168.0.40 0 -

108 01/23/1998 16:59:18 00:00:01 http 1882 80 192.168.1.30 192.168.0.40 0 -

109 01/23/1998 16:59:21 00:00:01 http 1883 80 192.168.1.30 192.168.0.40 0 -

110 01/23/1998 16:59:23 00:00:24 telnet 1884 23 192.168.1.30 192.168.0.20 1 guess

111 01/23/1998 16:59:24 00:00:01 http 1885 80 192.168.1.30 192.168.0.40 0 -

112 01/23/1998 16:59:26 00:00:02 http 1886 80 192.168.1.30 192.168.0.40 0 -

113 01/23/1998 16:59:29 00:00:02 http 1887 80 192.168.1.30 192.168.0.40 0 -

115 01/23/1998 16:59:33 00:00:02 http 1889 80 192.168.1.30 192.168.0.40 0 -
```

116 01/23/1998 16:59:33 00:01:41 telnet 1890 23 192.168.1.30 192.168.0.20 0 -

117 01/23/1998 16:59:36 00:00:02 http 1891 80 192.168.1.30 192.168.0.40 0 -

118 01/23/1998 16:59:36 00:00:12 ftp 1892 21 192.168.1.30 192.168.0.20 0 -

119 01/23/1998 16:59:42 00:00:00 ftp-data 20 1893 192.168.0.20 192.168.1.30 0 -

120 01/23/1998 16:59:45 00:00:01 ftp-data 20 1894 192.168.0.20 192.168.1.30 0 -

121 01/23/1998 16:59:47 00:00:00 ftp-data 20 1895 192.168.0.20 192.168.1.30 0 -

122 01/23/1998 16:59:53 00:00:01 smtp 1900 25 192.168.1.30 192.168.0.20 0 -

123 01/23/1998 16:59:57 00:00:16 ftp 43546 21 192.168.0.40 192.168.1.30 0 -

124 01/23/1998 17:00:01 00:00:00 ftp-data 20 43548 192.168.1.30 192.168.0.40 0 -

125 01/23/1998 17:00:02 00:00:02 rsh 1023 514 192.168.1.30 192.168.0.20 1 rcp

126 01/23/1998 17:00:03 00:00:22 telnet 1906 23 192.168.1.30 192.168.0.20 1 guess

127 01/23/1998 17:00:04 00:00:01 ftp-data 20 43550 192.168.1.30 192.168.0.40 0 -

128 01/23/1998 17:00:05 00:00:14 rlogin 1022 513 192.168.1.30 192.168.0.20 1 rlogin

129 01/23/1998 17:00:07 00:00:00 ftp-data 20 43552 192.168.1.30 192.168.0.40 0 -

130 01/23/1998 17:00:09 00:00:00 ftp-data 20 43554 192.168.1.30 192.168.0.40 0 -

131 01/23/1998 17:00:10 00:00:11 ftp 43555 21 192.168.0.40 192.168.1.30 0 -

132 01/23/1998 17:00:12 00:00:00 ftp-data 20 43558 192.168.1.30 192.168.0.40 0 -

133 01/23/1998 17:00:16 00:00:00 ftp-data 20 43562 192.168.1.30 192.168.0.40 0 -

The logs have been collected from DARPA's test run for intrusion detection systems. The sample contains simple attacks which are included to illustrate how intrusion detection systems will be scored. A session is labelled as containing an attack if it contains any component of an attack. Attacks include instances where a remote user illegally obtains local user-level privileges or local root-level privileges on a target machine and also

instances where a remote user surveys a potential target for weaknesses or searches for potential targets. Attacks in the sample data include the following:

Name	Description
guess	Remote user guesses many passwords to log into a target machine
ping-sweep	Low level ICMP ping sweep to identify target machines
port-scan	Determine which services on a target machine are active
phf	Run Unix command line on a web server
rlogin	Rlogin to target machine without a password
rsh	Execute a command on the target machine without a password
rcp	Remotely copy a file to/from target machine without a password

Another kind of parsing can be done to see if any changes to the database have been received. In that case, the following keywords are parsed for "DPI Rule: 1000608 - Generic SQL Injection Prevention".

PROXY ENCRYPTION LIBRARY

The proxy encryption library called the Nics Crypto Library was developed by David Nunez. This library was used by the ACDC^3 scheme.

The library has three main parts

- AFGHGlobalParameters.java
- AFGHProxyReEncryption.java
- ProxyMain.java

AFGHGLOBALPARAMETERS.JAVA

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package nics.crypto.proxy.afgh;

import it.unisa.dia.gas.jpbc.CurveGenerator;
import it.unisa.dia.gas.jpbc.CurveParameters;
import it.unisa.dia.gas.jpbc.Element;
import it.unisa.dia.gas.jpbc.ElementPowPreProcessing;
import it.unisa.dia.gas.jpbc.Field;
import it.unisa.dia.gas.jpbc.Pairing;
import it.unisa.dia.gas.plaf.jpbc.field.curve.CurveField;
import it.unisa.dia.gas.plaf.jpbc.pairing.DefaultCurveParameters;
import it.unisa.dia.gas.plaf.jpbc.pairing.a.TypeACurveGenerator;
import it.unisa.dia.gas.plaf.jpbc.pairing.a.TypeAPairing;
import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.InputStream;
import java.util.Random;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
```

```
 * @author david
 */
public class AFGHGlobalParameters {

    private int rBits, qBits;
    private Pairing e;
    private Field G1, G2, Zq;
    private Element g, Z;
    private ElementPowPreProcessing g_ppp, Z_ppp;

    private CurveParameters curveParams;
    private Random random;

    public AFGHGlobalParameters(DefaultCurveParameters curveParameters){
        initialize(curveParameters);
    }

    public AFGHGlobalParameters(int r, int q) {
        rBits = r;
        qBits = q;


        random = new Random(0);
        boolean generateCurveFieldGen = false;

        // Init the generator...
        CurveGenerator curveGenerator = new TypeACurveGenerator(random, rBits, qBits, generateCurveFieldGen);

        // Generate the parameters...
        curveParams = curveGenerator.generate();
        initialize(curveParams);
    }

    public AFGHGlobalParameters(InputStream is){
        curveParams = new DefaultCurveParameters();
        ((DefaultCurveParameters) curveParams).load(is);
        initialize(curveParams);
    }

    public AFGHGlobalParameters(File f) throws FileNotFoundException{
```

```
        this(new FileInputStream(f));
    }

    public AFGHGlobalParameters(byte[] b){
        this(new String(b));
    }

    public AFGHGlobalParameters(String cp){
        try {
            curveParams = new DefaultCurveParameters();

            ByteArrayInputStream is = new ByteArrayInputStream(cp.getBytes());
            ((DefaultCurveParameters) curveParams).load(is);
            initialize(curveParams);
        } catch (Exception ex) {
            Logger.getLogger(AFGHGlobalParameters.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    private void initialize(CurveParameters cp){
        random = new Random(0);
        //e = PairingFactory.getPairing(cp);

        e = new TypeAPairing(random, cp);

        // Groups G1 and G2 of prime order q
        G1 = e.getG1();
        G2 = e.getGT();



        // Field Zq
        Zq = e.getZr();



        // Global system parameters: g \in G1, Z = e(g,g) \in G2
        g = ((CurveField) G1).getGen().getImmutable();
//      if(g.isZero()){
//          System.out.println("g es 0!! :(");
```

```
//          System.exit(-1);
//      }
//      g = G1.newRandomElement().getImmutable();
      //System.out.println("g = " + ProxyMain.elementToString(g));


      Z = e.pairing(g, g).getImmutable();

      Z_ppp = Z.pow();
      g_ppp = g.pow();



      /*
      System.out.println(G1.getClass());
      System.out.println(G2.getClass());
      System.out.println(Zq.getClass());
      System.out.println(e.getClass());
      System.out.println(g.getClass());
      System.out.println(g.toBytes()[0]);
      System.out.println(Z.getClass());*/

   }

   public Field getG1() {
      return G1;
   }

   public Field getG2() {
      return G2;
   }

   public Element getZ() {
      return Z;
   }

   public Field getZq() {
      return Zq;
   }
```

```
public Pairing getE() {
    return e;
}

public Element getG() {
    return g;
}

public ElementPowPreProcessing getZ_ppp() {
    return Z_ppp;
}

public ElementPowPreProcessing getG_ppp() {
    return g_ppp;
}

@Override
public String toString() {
    return curveParams.toString();

    /*try {
        ByteArrayOutputStream os = new ByteArrayOutputStream();

        ObjectOutput oo = new ObjectOutputStream(os);
        curveParams.writeExternal(oo);

        os.close();
        return os.toString();
    } catch (IOException ex) {
        Logger.getLogger(GlobalParameters.class.getName()).log(Level.SEVERE, null, ex);
        return "";
    }*/
}

public byte[] toBytes() {
    return toString().getBytes();

    /*try {
        ByteArrayOutputStream os = new ByteArrayOutputStream();
```

```
        ObjectOutput oo = new ObjectOutputStream(os);
        curveParams.writeExternal(oo);

        os.close();
        return os.toString();
      } catch (IOException ex) {
        Logger.getLogger(GlobalParameters.class.getName()).log(Level.SEVERE, null, ex);
        return "";
      }*/
  }



}
```

AFGHPROXYREENCRYPTION.JAVA

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package nics.crypto.proxy.afgh;

import it.unisa.dia.gas.jpbc.Element;
import it.unisa.dia.gas.jpbc.ElementPowPreProcessing;
import it.unisa.dia.gas.jpbc.Field;
import it.unisa.dia.gas.jpbc.Pairing;
import it.unisa.dia.gas.jpbc.PairingPreProcessing;
import java.util.Arrays;
import nics.crypto.Tuple;

/**
 *
 * @author david
 */
public class AFGHProxyReEncryption {

   public static Element generateSecretKey(AFGHGlobalParameters global) {
```

```
        Field Zq = global.getZq();

        /*
         * KEY GENERATION
         */

        // sk = a \in Zq
        return Zq.newRandomElement().getImmutable();
    }

//    public static byte[] generateSecretKey(GlobalParameters global) {
//        return generateSecretKey(global).toBytes();
//    }
    public static Element generatePublicKey(Element sk, AFGHGlobalParameters global) {

        ElementPowPreProcessing g = global.getG_ppp();

        // pk = g^sk
        return g.powZn(sk).getImmutable();
    }

    public static byte[] generatePublicKey(byte[] sk_bytes, AFGHGlobalParameters global) {

        Element sk = bytesToElement(sk_bytes, global.getZq());

        return generatePublicKey(sk, global).toBytes();
    }

    public static Element generateReEncryptionKey(Element pk_b, Element sk_a) {

        /*
         * Re-Encryption Key Generation
         */

        // RK(a->b) = pk_b ^(1/sk_a) = g^(b/a)
        Element rk_a_b = pk_b.powZn(sk_a.invert());
        return rk_a_b.getImmutable();

    }
```

```
public static byte[] generateReEncryptionKey(byte[] pk_bytes, byte[] sk_bytes, AFGHGlobalParameters global) {
    return generateReEncryptionKey(
            bytesToElement(pk_bytes, global.getG1()),
            bytesToElement(sk_bytes, global.getZq())).toBytes();
}

public static byte[] firstLevelEncryption(byte[] message, byte[] pk_a, AFGHGlobalParameters global) {

    Field G2 = global.getG2();
    Field G1 = global.getG1();

    // message = m \in G2
    Element m = bytesToElement(message, G2);

    // pk_a \in G1
    Element pk = bytesToElement(pk_a, G1);

    Tuple c = firstLevelEncryption(m, pk, global);

    return mergeByteArrays(c.get(1).toBytes(), c.get(2).toBytes());

}

public static Tuple firstLevelEncryption(Element m, Element pk_a, AFGHGlobalParameters global) {

    /*
     * First Level Encryption
     * c = (c1, c2)     c1, c2 \in G2
     *      c1 = Z^ak = e(g,g)^ak = e(g^a,g^k) = e(pk_a, g^k)
     *      c2 = m·Z^k
     */

    Field G2 = global.getG2();
    Field Zq = global.getZq();

    Pairing e = global.getE();

    Element Z = global.getZ();
    Element g = global.getG();
```

```
        // random k \in Zq
        Element k = Zq.newRandomElement().getImmutable();

        // g^k
        Element g_k = g.powZn(k);

        // c1 = Z^ak = e(g,g)^ak = e(g^a,g^k) = e(pk_a, g^k)
        Element c1 = e.pairing(pk_a, g_k);


        // c2 = m·Z^k
        Element c2 = m.mul(Z.powZn(k));


        // c = (c1, c2)

        Tuple c = new Tuple(c1, c2);

        return c;

    }

    public static byte[] secondLevelEncryption(byte[] message, byte[] pk_a, AFGHGlobalParameters global) {

        Field G2 = global.getG2();
        Field G1 = global.getG1();

        System.out.println(G2.getClass());

        System.out.println("G2: " + G2.getLengthInBytes());
        // message = m \in G2
        Element m = bytesToElement(message, G2);
//      System.out.println("M : " + Arrays.toString(m.toBytes()));
        // pk_a \in G1
        Element pk = bytesToElement(pk_a, G1);



        Tuple c = secondLevelEncryption(m, pk, global);
```

```
    return mergeByteArrays(c.get(1).toBytes(), c.get(2).toBytes());

}

public static Tuple secondLevelEncryption(Element m, Element pk_a, AFGHGlobalParameters global) {

    /*
     * Second Level Encryption
     * c = (c1, c2)     c1 \in G1, c2 \in G2
     *      c1 = g^ak = pk_a^k
     *      c2 = m·Z^k
     */

    //Field G2 = global.getG2();
    Field Zq = global.getZq();

    Pairing e = global.getE();

    Element Z = global.getZ();



    // random k \in Zq
    Element k = Zq.newRandomElement().getImmutable();
    //System.out.println("k = " + elementToString(k));

    // c1 = pk_a^k
    Element c1 = pk_a.powZn(k).getImmutable();


    // c2 = m·Z^k
    Element c2 = m.mul(Z.powZn(k)).getImmutable();



    // c = (c1, c2)
    Tuple c = new Tuple(c1, c2);
```

```
        return c;

    }


    public static Tuple secondLevelEncryption(Element m, ElementPowPreProcessing pk_a_PPP,
AFGHGlobalParameters global) {

        /*
         * Second Level Encryption
         * c = (c1, c2)     c1 \in G1, c2 \in G2
         *      c1 = g^ak = pk_a^k
         *      c2 = m·Z^k
         */

        //Field G2 = global.getG2();
        Field Zq = global.getZq();

        Pairing e = global.getE();

        //Element Z = global.getZ();

        ElementPowPreProcessing Z_PPP = global.getZ_ppp();



        // random k \in Zq
        Element k = Zq.newRandomElement().getImmutable();
        //System.out.println("k = " + elementToString(k));

        // c1 = pk_a^k
        Element c1 = pk_a_PPP.powZn(k).getImmutable();


        // c2 = m·Z^k
        Element c2 = m.mul(Z_PPP.powZn(k)).getImmutable();
```

```
    // c = (c1, c2)
    Tuple c = new Tuple(c1, c2);

    return c;

}

public static Tuple reEncryption(Tuple c, Element rk, AFGHGlobalParameters global) {

    /*
     * Re-Encryption
     * c' = ( e(c1, rk) , c2)   \in G2 x G2
     */

    Pairing e = global.getE();



    return new Tuple(e.pairing(c.get(1), rk), c.get(2));

}

 public static Tuple reEncryption(Tuple c, Element rk, PairingPreProcessing e_ppp) {

    /*
     * Re-Encryption
     * c' = ( e(c1, rk) , c2)   \in G2 x G2
     */

    return new Tuple(e_ppp.pairing(c.get(1)), c.get(2));

}

public static byte[] reEncryption(byte[] c, byte[] rk, AFGHGlobalParameters global) {
    //System.out.println("R: " + Arrays.toString(c));
    // c1 \in G1, c2 \in G2
    Field G1 = global.getG1();
    Field G2 = global.getG2();

    Element c1 = G1.newElement();
```

```
        int offset = bytesToElement(c, c1, 0);
        c1 = c1.getImmutable();

        Element c2 = G2.newElement();
        bytesToElement(c, c2, offset);
        c2 = c2.getImmutable();


        Tuple t = reEncryption(new Tuple(c1, c2), bytesToElement(rk, G1), global);

        return mergeByteArrays(t.get(1).toBytes(), t.get(2).toBytes());

    }

    public static Element firstLevelDecryption(Tuple c, Element sk, AFGHGlobalParameters global) {
        // c1, c2 \in G2
        Element alpha = c.get(1);
        Element beta = c.get(2);

        Element sk_inverse = sk.invert();

        Element m = beta.div(alpha.powZn(sk_inverse));

        return m;
    }

    public static Element firstLevelDecryptionPreProcessing(Tuple c, Element sk_inverse, AFGHGlobalParameters
global) {
        // c1, c2 \in G2
        Element alpha = c.get(1);
        Element beta = c.get(2);

        Element m = beta.div(alpha.powZn(sk_inverse));

        return m;
    }

    public static byte[] firstLevelDecryption(byte[] b, byte[] sk, AFGHGlobalParameters global) {
        //System.out.println(Arrays.toString(b));
```

```
        // c1, c2 \in G2
        Field G2 = global.getG2();

        Element alpha = G2.newElement();
        int offset = bytesToElement(b, alpha, 0);
        alpha = alpha.getImmutable();

        Element beta = G2.newElement();
        bytesToElement(b, beta, offset);
        beta = beta.getImmutable();

        //System.out.println(Arrays.toString(beta.toBytes()));



        Element key = bytesToElement(sk, global.getZq());

//      key.invert();
//      System.out.println(Arrays.toString(key.invert().toBytes()));

        Element m = firstLevelDecryption(new Tuple(alpha, beta), key, global);

        return m.toBytes();
    }

    public static byte[] secondLevelDecryption(byte[] b, byte[] sk, AFGHGlobalParameters global) {
        // c1 \in G1, c2 \in G2
        Field G1 = global.getG1();
        Field G2 = global.getG2();

        Element alpha = G1.newElement();
        int offset = bytesToElement(b, alpha, 0);
        alpha = alpha.getImmutable();

        Element beta = G2.newElement();
        bytesToElement(b, beta, offset);
        beta = beta.getImmutable();

        Element key = bytesToElement(sk, global.getZq());
```

```
    Element m = secondLevelDecryption(new Tuple(alpha, beta), key, global);

    return m.toBytes();

}

public static Element secondLevelDecryption(Tuple c, Element sk, AFGHGlobalParameters global) {

    Element alpha = c.get(1);
    Element beta = c.get(2);

    Pairing e = global.getE();
    Element g = global.getG();

    Element m = beta.div(e.pairing(alpha, g).powZn(sk.invert()));

    return m;
}

public static Element decryption(Tuple c, Element sk, AFGHGlobalParameters global) {
    Field G2 = global.getG2();

    // if c1 \in G2 then First-Level
    if (c.get(1).getField().equals(G2)) {
        return firstLevelDecryption(c, sk, global);
    } else {
        return secondLevelDecryption(c, sk, global);
    }
}

public static Element stringToElement(String s, Field G) {
    //System.out.println(s + " = " + Arrays.toString(s.getBytes()));
    //return bytesToElement(Base64.decode(s), G);
    return bytesToElement(s.getBytes(), G);
}

public static Element bytesToElement(byte[] b, Field G) {
    int maxLengthBytes = G.getLengthInBytes();
```

```
    //System.out.println("maxLengthBytes = " + maxLengthBytes);
    if (b.length > maxLengthBytes) {
        throw new IllegalArgumentException("Input must be less than " + maxLengthBytes + " bytes");
    }
    //System.out.println(Arrays.asList(b));

    Element x = G.newElement();
    x.setFromBytes(b);

    //Element x = G.newElement(new BigInteger(1, b));
    return x.getImmutable();
}

public static int bytesToElement(byte[] b, Element x, int offset) {



    offset += x.setFromBytes(b, offset);

    return offset;
}

public static String elementToString(Element x) {
    //return Base64.encodeBytes(x.toBytes());
    return new String(x.toBytes()).trim();
}

public static byte[] mergeByteArrays(byte[]... bs) {
    int newLength = 0;
    for (byte[] b : bs) {
        newLength += b.length;
    }

    byte[] merge = new byte[newLength];

    int from = 0;
    for (byte[] b : bs) {
        System.arraycopy(b, 0, merge, from, b.length);
        from += b.length;
    }
```

```
        return merge;
    }
}
```

PROXYMAIN.JAVA

```
*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package nics.crypto.proxy.afgh;

import nics.crypto.Tuple;
import it.unisa.dia.gas.jpbc.*;
/**
 *
 * @author david
 */
public class ProxyMain {

    static long cpuTime;
    static long time[] = new long[20];
    static int i = 0;
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) throws Exception {



        //java.security.

        cpuTime = System.nanoTime();

        // 80 bits seg: r = 160, q = 512
        // 128 bits seg: r = 256, q = 1536
        // 256 bits seg: r = 512, q = 7680
```

```
        int rBits = 256; //160;    // 20 bytes
        int qBits = 1536; //512;    // 64 bytes

        AFGHGlobalParameters global = new AFGHGlobalParameters(rBits, qBits);

        medirTiempoMicroSegundos();

//      // Secret keys
//
//      byte[] sk_a = AFGH.generateSecretKey(global).toBytes();
//
//      System.out.println(medirTiempo());
//
//      byte[] sk_b = AFGH.generateSecretKey(global).toBytes();
//
//      System.out.println(medirTiempo());
//
//      // Public keys
//
//      byte[] pk_a = AFGH.generatePublicKey(sk_a, global);
//
//      System.out.println(medirTiempo());
//
//      byte[] pk_b = AFGH.generatePublicKey(sk_b, global);
//
//      System.out.println(medirTiempo());
//
//      // Re-Encryption Key
//
//      byte[] rk_a_b = AFGH.generateReEncryptionKey(pk_b, sk_a, global);
//
//      System.out.println(medirTiempo());
//
//      String message = "David";
//      byte[] m = message.getBytes();
//
//      System.out.println(medirTiempo());
//
//      byte[] c_a = AFGH.secondLevelEncryption(m, pk_a, global);
//
```

```
//      System.out.println(medirTiempo());
//
//      String c_a_base64 = Base64.encodeBase64URLSafeString(c_a);
//      //System.out.println("c_a_base64 = " + c_a_base64);
//
//      System.out.println(medirTiempo());
//
//      String rk_base64 = Base64.encodeBase64URLSafeString(rk_a_b);
//      //System.out.println("rk_base64 = " + rk_base64);
//      System.out.println(medirTiempo());
//
//      byte[] c, rk;
//      rk = Base64.decodeBase64(rk_base64);
//
//      System.out.println(medirTiempo());
//
//      c = Base64.decodeBase64(c_a_base64);
//
//      System.out.println(medirTiempo());
//
//      byte[] c_b = AFGH.reEncryption(c, rk, global);
//      //System.out.println("cb: " + Arrays.toString(c_b));
//      System.out.println(medirTiempo());
//
//      String c_b_base64 = Base64.encodeBase64URLSafeString(c_b);
//      //System.out.println("c_b_base64 = " + c_b_base64);
//
//      System.out.println(medirTiempo());
//
//      c = Base64.decodeBase64(c_b_base64);
//
//      System.out.println(medirTiempo());
//
//      byte[] m2 = AFGH.firstLevelDecryption(c_b, sk_b, global);
//      //System.out.println("m2:" + new String(m2));
//
//      System.out.println(medirTiempo());
//
//      assert message.equals(new String(m2).trim());
//
```

```
//      System.out.println();
//      System.out.println(global.toBytes().length);
//      System.out.println(sk_a.length);
//      System.out.println(sk_b.length);
//      System.out.println(pk_a.length);
//      System.out.println(pk_b.length);
//      System.out.println(rk_a_b.length);
//      System.out.println(m.length);
//      System.out.println(c_a.length);
//      System.out.println(c_b.length);
//
//      //
//      Map<String, byte[]> map = new HashMap<String, byte[]>();
//      map.put("sk_a", sk_a);
//      map.put("sk_b", sk_b);
//      map.put("pk_a", pk_a);
//      map.put("pk_b", pk_b);
//      map.put("rk_a_b", rk_a_b);
//      map.put("global", global.toBytes());
//      map.put("c_a_base64", c_a_base64.getBytes());
//
//      ObjectOutputStream fos = new ObjectOutputStream(new
FileOutputStream("/Users/david/Desktop/pre.object"));
//      fos.writeObject(map);
//      fos.close();
      //

      // Secret keys

      Element sk_a = AFGHProxyReEncryption.generateSecretKey(global);

      medirTiempoMicroSegundos();

      Element sk_b = AFGHProxyReEncryption.generateSecretKey(global);

      medirTiempoMicroSegundos();

      Element sk_b_inverse = sk_b.invert();

      medirTiempoMicroSegundos();
```

```
// Public keys

Element pk_a = AFGHProxyReEncryption.generatePublicKey(sk_a, global);

medirTiempoMicroSegundos();

Element pk_b = AFGHProxyReEncryption.generatePublicKey(sk_b, global);

medirTiempoMicroSegundos();

ElementPowPreProcessing pk_a_ppp = pk_a.pow();

medirTiempoMicroSegundos();

// Re-Encryption Key

Element rk_a_b = AFGHProxyReEncryption.generateReEncryptionKey(pk_b, sk_a);

medirTiempoMicroSegundos();

String message = "12345678901234567890123456789012";
Element m = AFGHProxyReEncryption.stringToElement(message, global.getG2());

medirTiempoMicroSegundos();

Tuple c_a = AFGHProxyReEncryption.secondLevelEncryption(m, pk_a_ppp, global);

medirTiempoMicroSegundos();

PairingPreProcessing e_ppp = global.getE().pairing(rk_a_b);

medirTiempoMicroSegundos();

Tuple c_b = AFGHProxyReEncryption.reEncryption(c_a, rk_a_b, e_ppp);

medirTiempoMicroSegundos();

Element m2 = AFGHProxyReEncryption.firstLevelDecryptionPreProcessing(c_b, sk_b_inverse, global);
```

```
        medirTiempoMicroSegundos();

        assert message.equals(new String(m2.toBytes()).trim());

        for(int j = 0; j < i; j++){
           System.out.println(time[j]);
        }

//      System.out.println("m string : " + message.getBytes().length);
//      System.out.println("m in G2 : " + m.toBytes().length);
//      System.out.println("c_a_1 in G2: " + c_a.get(1).toBytes().length);
//      System.out.println("c_a_2 in G1: " + c_a.get(2).toBytes().length);
//      System.out.println("c_b_1 in G2: " + c_b.get(1).toBytes().length);
//      System.out.println("c_b_2 in G2: " + c_b.get(2).toBytes().length);
//      System.out.println("m2 in G2 : " + m2.toBytes().length);
        //System.out.println(AFGH.elementToString(m2));

        //System.out.println(medirTiempo());

    }



    public static long medirTiempoMicroSegundos() {
        time[i] = (System.nanoTime() - cpuTime)/1000;
        i++;
        cpuTime = System.nanoTime();
        return time[i];
    }
}
```

CODE FOR THE ACDC3 SCHEME

The following code was developed for testing the ACDC^3 scheme.

```
package nics.crypto.proxy.afgh;
import nics.crypto.Tuple;
import it.unisa.dia.gas.jpbc.*;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.util.ArrayList;
import org.apache.commons.io.FileUtils;

public class Main {

        public static final int rBits = 256; //160;    // 20 bytes
        public static final int qBits = 1536; //512;    // 64 bytes
        public static final GlobalParameters global = new GlobalParameters(rBits, qBits);

        public static void main(String[] args) throws IOException, ClassNotFoundException
        {
                // Secret keys

    Element sk_a = AFGH.generateSecretKey(global);

    Element sk_b = AFGH.generateSecretKey(global);

    // Public keys

    Element pk_a = AFGH.generatePublicKey(sk_a, global);
```

```
Element pk_b = AFGH.generatePublicKey(sk_b, global);

ElementPowPreProcessing pk_a_ppp = pk_a.pow();

// Re-Encryption Key

Element rk_a_b = AFGH.generateReEncryptionKey(pk_b, sk_a);

// Plain Text

File plainText = new File("sheffieldlogosmall.png");

log(plainText.getAbsolutePath());

byte[] b = FileUtils.readFileToByteArray(plainText);

Element m = AFGH.bytesToElement(b, global.getG2());

Tuple CT = AFGH.secondLevelEncryption(m, pk_a_ppp, global);

PairingPreProcessing e_ppp = global.getE().pairing(rk_a_b);

// Re-Encryption into ciphertext CT

Tuple CTT = AFGH.reEncryption(CT, rk_a_b, e_ppp);


// Send to TTP

sendToTTP(CTT);

// Decryption by Bob
```

```
        Tuple CTTT = readFromTTP("CTT.ser");

        Element sk_b_inverse = sk_b.invert();

        Element mResult = AFGH.firstLevelDecryptionPreProcessing(CTTT, sk_b_inverse, global);

        FileUtils.writeByteArrayToFile(new File("result.txt"), mResult.toBytes());
            }

            private static void log(String str)
            {
                    System.out.println(str);
            }

            private static void sendToTTP(Tuple t) throws IOException
            {
                    ArrayList<byte[]> res = t.toBytes();
                    FileOutputStream fout = new FileOutputStream("CTT.ser");
        ObjectOutputStream oos = new ObjectOutputStream(fout);
        oos.writeObject(res);
        oos.flush();
        oos.close();
        fout.flush();
        fout.close();
            }

            private static Tuple readFromTTP(String file) throws IOException,
ClassNotFoundException
            {
                    FileInputStream fin = new FileInputStream(file);
        ObjectInputStream ois = new ObjectInputStream(fin);
        @SuppressWarnings("unchecked")
```

```
                    ArrayList<byte[]> CTTList = new ArrayList<byte[]>( (ArrayList<byte[]>)
ois.readObject());
        ois.close();
        fin.close();

        Element[] elements = new Element[CTTList.size()];

        for (int i = 0; i < CTTList.size(); i++)
        {
            elements[i] = AFGH.bytesToElement(CTTList.get(i), global.getG2());
        }
        return new Tuple(elements);
            }
}
```

RESEARCH PLAN TIMELINE AND MILESTONES

A Gantt chart shown in figure 35 entails the major milestones and timeline of the project.

ID	Task Name	Start	Finish	Duration
1	Development of new scheme (RG1 and RG4)	01/08/2011	28/10/2011	65d
2	ACDC3 Scheme (RG1 and RG4)	01/11/2011	01/02/2012	67d
3	Progress Report Submission (Milestone)	11/10/2011	31/01/2012	81d
4	Redesigning ACDC3 Scheme and countering shortcomings	11/10/2011	01/02/2012	82d
5	Developing Risk Assessment Scheme	03/02/2012	11/05/2012	71d
6	Design Scheme ACDC3 (RG1 and RG4)	14/05/2012	13/07/2012	45d
7	Access Control and Data Confidentiality in Cloud Computing (EU Patent Filed)	15/02/2012	13/04/2012	43d
8	Using Corus methodology for risk assessment scheme	16/07/2012	14/09/2012	45d
9	Thesis outline (Milestone)	01/02/2012	29/06/2012	108d
10	Security Threat Analysis (Research Paper, Cloudcom 2012)	01/08/2012	30/11/2012	88d
11	Secure Scalable Video Paper (1st Draft)	17/09/2012	14/11/2012	43d
12	Deployment on OPTIMIS testbed (Risk Assessment Scheme)	16/11/2012	14/12/2012	21d
13	Implementing ACDC3 Scheme (RG1 and RG4)	13/12/2012	27/03/2013	75d
14	Secure Scalable Video Research paper writeup	01/02/2013	01/03/2013	21d
15	Thesis Audit (Milestone)	05/11/2012	28/12/2012	40d
16	Integration with OPTIMIS toolkit Security Risk Framework	28/03/2013	24/05/2013	42d
17	Publishing of Scalable Video Research Paper	18/09/2013	20/06/2014	198d
18	Thesis Submission First Draft	23/06/2014	21/08/2014	44d
19	Thesis Submission Second Draft	22/08/2014	03/11/2014	52d
20	Thesis Seminar (Milestone)	01/04/2013	31/07/2013	88d
21	Final Thesis Submission (Milestone)	01/10/2013	31/10/2013	23d

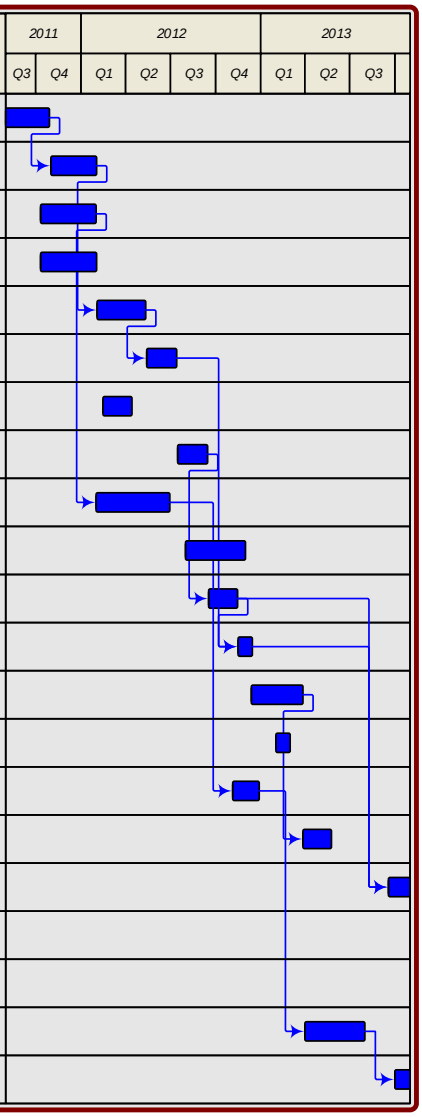

Figure 35: Gantt chart of the project

CLAIMS AND ABSTRACT OF THE PATENT (ACDC^3)

The above research work was filed by the BT IP Department in the form of two patents to the EU Patent Office and US Patent Office. BT's reference number for the patent is Europe A32311 [169] [170].

ABSTRACT

Embodiments of the invention provide a method and system which allow for ready revocation of end user access rights by virtue of storing data in an encrypted form in a network environment, and using a trusted proxy server to re-encrypt the data itself to permit eventual decryption of the data by an authorised end user. However, if the end user's access rights are revoked then the trusted proxy does not perform the re-encryption of the data, and the end user is not then able to subsequently decrypt data stored in the network environment, even if it is able to access the data without permission. Embodiments therefore have advantages that access control is decoupled from data confidentiality to provide scalability, and revocation of user access rights can be accomplished without requiring re-encryption of the stored data.

CLAIMS

Following are the claims of the patent,

1. A method for use in accessing data from network data storage, the data being encrypted with one or more layers of encryption including a first encryption layer applied by the data owner, the method comprising:

receiving a request from a data consumer for access to data stored in the network data storage;

determining whether to grant the request in dependence on whether the data consumer has access rights to the requested data;

obtaining a proxy re-encryption key generated by the data owner; and

if it is determined that the data consumer may access the data, obtaining the requested data from the network data storage and proxy re-encrypting the data to enable subsequent decryption of the first encryption layer applied by the data owner whereby to enable eventual access to the data.

2. A method according to claim 1, wherein the re-encryption key re-encrypts the data so that the first encryption layer may be decrypted by the data consumer, the method further comprising sending the re-encrypted data to the data consumer.

3. A method according to claim 2, wherein the data has a single layer of encryption being the first layer, wherein the data consumer is able to decrypt the re-encrypted data to plaintext data to access the data.

4. A method according to claim 1, wherein the re-encryption key re-encrypts the data so that the first encryption layer may be decrypted by the trusted authority, the method further comprising, at the trusted authority, decrypting the first encryption layer.

5. A method according to claim 4, wherein the data has at least two layers of encryption, being one or more other layers and the first layer, the decryption resulting in the data encrypted with the one or more other layers.

6. A method according to claims 4 or 5, and further comprising sending the proxy decrypted data to the data consumer, the data consumer then obtaining the decryption key to decrypt the one or more first layers to obtain plaintext data from the data owner.

7. A method according to claim 6, wherein the trusted authority requests the decryption key to decrypt the one or more first layers from the data owner, and forwards the decryption key to the data consumer.

8. A method for use in storing data in network data storage, the method comprising:

encrypting data to be stored in the network data storage with one or more layers of encryption, including at least a first encryption layer;

storing the encrypted data in the network data storage;

generating a proxy re-encryption key to allow a trusted authority to re-encrypt data encrypted with the first encryption layer so that the first encryption layer may be decrypted by a third party; and

sending the proxy re-encryption key to the trusted authority.

9. A method according to claim 8, wherein the re-encryption key is generated so as to be able to re-encrypt the data such that the first encryption layer may be decrypted by the data consumer.

10. A method according to claim 9, wherein the data has a single layer of encryption being the first layer, wherein the data consumer is able to decrypt the re-encrypted data to plaintext data to access the data.

11. A method according to claim 8, wherein the re-encryption key re-encrypts the data so that the first encryption layer may be decrypted by the trusted authority.

12. A method according to claim 10, wherein the data has at least two layers of encryption, being one or more other layers and the first layer, the method further comprising, receiving a request for the decryption key or keys for the one or more other layers, and sending the keys in response to the request.

13. A computer program or suite of computer programs so arranged such that when executed by a computer system it/they cause(s) the computer system to operate in accordance with the method of any of the preceding claims.

14. A computer readable medium storing a computer program or at least one of a suite of computer programs according to claim 13.

15. A system, comprising:

at least one processor;

memory; and

at least one computer readable medium storing a computer program or suite of computer programs so arranged such that when loaded into memory and executed by the processor they cause the system to operate in accordance with the method of any of claims 1 to 12.

REFERENCES

[1] L. Rodero-Merino, M. Lindner, J. Caceres, and L. M. Vaquero, “A break in the clouds,” *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1. p. 50, 2008.

[2] R. Buyya, “Market-Oriented Cloud Computing: Vision, Hype, and Reality of Delivering Computing as the 5th Utility,” *2009 9th IEEE/ACM Int. Symp. Clust. Comput. Grid*, 2009.

[3] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility,” *Futur. Gener. Comput. Syst.*, vol. 25, no. 6, p. 17, 2009.

[4] Siemens IT solutions, “Cloud Computing – Business Models, Value Creation Dynamics and Advantages for Customers,” 2010. [Online]. Available: http://www.it-solutions.siemens.com/b2b/it/en/global/Documents/Publications/CloudComputing_Whitepaper_PDF_e.pdf. [Accessed: 01-Nov-2010].

[5] European research consortium for Informatics and mathematics (ERCIM), *Cloud Computing: Special theme*. 2010.

[6] EU, “OPTIMIS EU project FP7,” 2010. [Online]. Available: http://www.optimis-project.eu/. [Accessed: 20-Mar-2012].

[7] BT, “Virtual Data Centre (VDC) – A New Concept in Service Delivery,” 2010. [Online]. Available: http://globalservices.bt.com/LeafAction.do?Record=Virtual_Data_Centre_products_uk_en-gb. [Accessed: 24-Jun-2012].

[8] A. J. Ferrer, F. Hernández, J. Tordsson, E. Elmroth, A. Ali-Eldin, C. Zsigri, R. Sirvent, J. Guitart, R. M. Badia, K. Djemame, W. Ziegler, T. Dimitrakos, S. K. Nair, G. Kousiouris, K. Konstanteli, T. Varvarigou, B. Hudzia, A. Kipp, S. Wesner, M. Corrales, N. Forgó, T. Sharif, and C. Sheridan, “OPTIMIS: A holistic approach to cloud service provisioning,” *Futur. Gener. Comput. Syst.*, vol. 28, no. 1, pp. 66–77, 2012.

[9] I. Hogganvik, F. Vraalsen, F. Braber, K. Stølen, and M. S. Lund, “Model-based security analysis in seven steps — a guided tour to the CORAS method,” *BT Technology Journal*, vol. 25, no. 1. pp. 101–117, 2007.

[10] F. D. Braber, “The Coras Model-based method for security risk analysis,” *SIntef*, 2006.

[11] R. Fredriksen, M. Kristiansen, B. Gran, and K. Stolen, “The CORAS framework for a model-based risk management process,” in *Lecture notes in*, vol. 2434, S. Anderson, M. Felici, and S. Bologna, Eds. Springer-Verlag, 2002, pp. 94–105.

[12] Information Security Forum (ISF), “Information risk analysis methodology (IRAM),” 2010. [Online]. Available: https://www.securityforum.org/iram#iramtva. [Accessed: 15-Apr-2012].

[13] R. Sandhu and J. Park, “Usage Control: A Vision for Next Generation Access Control,” 2003. [Online]. Available: http://profsandhu.com/confrnc/misconf/2003_MMS_UCON.pdf. [Accessed: 08-Aug-2013].

[14] I. Foster, Y. Z. Y. Zhao, I. Raicu, and S. Lu, “Cloud Computing and Grid Computing 360-Degree Compared,” *2008 Grid Comput. Environ. Work.*, 2008.

[15] P. Mell and T. Grance, “The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology,” 2011.

[16] D. F. Ferraiolo, D. R. Kuhn, R. Sandhu, S. Gavrila, and R. Chandramouli, “Proposed NIST standard for role-based access control,” *ACM Transactions on Information and System Security*, vol. 4, no. 3. pp. 224–274, 2001.

[17] L. Youseff, M. Butrico, and D. Da Silva, “Toward a Unified Ontology of Cloud Computing: Grid Computing Environments Workshop, 2008. GCE ’08,” *Grid Comput. Environ. Work. 2008. GCE ’08*, pp. 1–10, 2008.

[18] F. Shimba, “Cloud Computing:Strategies for Cloud Computing Adoption,” Dublin Institute of Technology, 2010.

[19] A. Khajeh-Hosseini, D. Greenwood, J. W. Smith, and I. Sommerville, “The Cloud Adoption Toolkit: supporting cloud adoption decisions in the enterprise.,” *Softw., Pr. Exper.*, vol. 42, no. 4, pp. 447–465, 2012.

[20] A. Khajeh-Hosseini, I. Sommerville, and I. Sriram, “Research Challenges for Enterprise Cloud Computing,” *CoRR*, vol. abs/1001.3, 2010.

[21] S. Migay and M. Govikar, “Does Cloud Computing Have a ‘Green’ Lining?,” *Gartner Inc*, 2010.

[22] J. Staten, “Is Cloud Computing Ready For The Enterprise ?,” *Reproduction*, vol. 7, p. 15, 2008.

[23] W. Fellows, “Partly Cloudy, Blue-Sky Thinking About Cloud Computing,” *WhitePaper, 451 Group*, 2008.

[24] R. Buyya and K. Bubendorfer, *Market-oriented grid and utility computing, Ch12 SLA-BASED RESOURCE MANAGEMENT AND ALLOCATION*. Wiley Online Library, 2009.

[25] Amazon, “Amazon Elastic Compute Cloud (EC2),” *AWS*. [Online]. Available: http://aws.amazon.com/ec2/. [Accessed: 06-May-2013].

[26] Google, “Google Compute.” [Online]. Available: https://cloud.google.com/compute/. [Accessed: 11-Nov-2011].

[27] IBM, “IBM IaaS.” [Online]. Available: http://www.ibm.com/cloud-computing/us/en/iaas.html. [Accessed: 11-Nov-2014].

[28] Google, “Google Apps Engine,” *Google*. [Online]. Available: http://code.google.com/appengine/. [Accessed: 06-May-2011].

[29] VmWare, “Vsphere,” *VmWare*. [Online]. Available: http://www.vmware.com/products/vsphere/mid-size-and-enterprise-business/features.html. [Accessed: 12-Jun-2011].

[30] Openstack, “OpenStack.” [Online]. Available: http://www.openstack.org/. [Accessed: 11-Nov-2014].

[31] “HP Cloud.” [Online]. Available: http://www.hpcloud.com/. [Accessed: 07-Jan-2015].

[32] S. K. Nair, S. Porwal, T. Dimitrakos, A. J. Ferrer, J. Tordsson, T. Sharif, C. Sheridan, M. Rajarajan, and A. U. Khan, “Towards Secure Cloud Bursting, Brokerage and Aggregation,” *2010 Eighth IEEE Eur. Conf. Web Serv.*, pp. 189–196, 2010.

[33] NVIDIA, “NVIDIA Reality Server,” *NVIDIA*. [Online]. Available: http://www.nvidia.com/object/realityserver.html. [Accessed: 09-Apr-2011].

[34] AWS, “Amazon Simple Storage Service,” *Amazon*. [Online]. Available: http://aws.amazon.com/s3/. [Accessed: 15-Jul-2011].

[35] P. Mell and T. Grance, “The NIST Definition of Cloud Computing: Recommendations of the National Institute of Standarts and Technology,” *Natl. Inst. Stand. Technol.*, vol. 800–145, pp. 1–7, 2011.

[36] AWS, “Amazon Web Services,” *Overview of Secuirty Processes*. [Online]. Available: http://s3.amazonaws.com/aws_blog/AWS_Security_Whitepaper_2008_09.pdf. [Accessed: 22-Jul-2011].

[37] Rackspace, “Rackspace security data sheet,” 2011. [Online]. Available: http://broadcast.rackspace.com/downloads/pdfs/RackspaceSecurityApproach.pdf. [Accessed: 22-Jul-2011].

[38] Rackspace, “Rackspace,” *Rackspace*. [Online]. Available: http://www.rackspace.co.uk/rackspace-home/. [Accessed: 22-Jul-2011].

[39] GoGrid, “GoGrid website,” *GoGrid*. [Online]. Available: http://www.gogrid.com/. [Accessed: 21-Jul-2011].

[40] Terremark, “Terremark website,” *Terremark*. [Online]. Available: http://www.terremark.com/default.aspx. [Accessed: 14-Jul-2011].

[41] GoGrid, “GoGrid Security Data Sheet,” *Security data sheet*. [Online]. Available: http://storage.pardot.com/3442/33711/GoGrid_hosted_private_cloud_data_sheet_20110111.pdf. [Accessed: 21-Jul-2011].

[42] Joyent, “Joyent Security architecture,” *Joyent*. [Online]. Available: http://www.joyent.com/software/smartdatacenter/security-architecture/. [Accessed: 22-Jul-2012].

[43] Xen, “Xen Virtualization Engine,” *Xen*. [Online]. Available: http://www.xen.org/. [Accessed: 10-Jul-2011].

[44] SAS, “SAS 70 Type 2 Audit,” 2012. [Online]. Available: http://sas70.com/sas70_overview.html. [Accessed: 09-Nov-2011].

[45] Windows, “Windows Azure.” [Online]. Available: http://www.microsoft.com/windowsazure/. [Accessed: 22-Jun-2011].

[46] Force.com, “Force.com.” [Online]. Available: http://www.salesforce.com/platform/. [Accessed: 25-Jul-2011].

[47] Force.com, “Force.com,” *Security Overview*. [Online]. Available: http://wiki.developerforce.com/index.php/An_Overview_of_Force.com_Security. [Accessed: 25-Jul-2011].

[48] Heroku, “Heroku.” [Online]. Available: http://heroku.com/. [Accessed: 29-Jul-2011].

[49] AWS, “Amazon Web-Services (AWS),” *Amazon*. [Online]. Available: http://aws.amazon.com/. [Accessed: 22-Jun-2011].

[50] N. Fips, “197: Announcing the advanced encryption standard (AES),” *… Technol. Lab. Natl. Inst. Stand. …*, vol. 2009, pp. 8–12, 2001.

[51] O. Hart, “Regulation and Sarbanes-Oxley,” *J. Account. Res.*, vol. 47, no. 2, pp. 437–445, 2009.

[52] T. H. E. Data, P. Act, D. Protection, and T. Act, “Data Protection Act,” *UK Legislation*, 1998. [Online]. Available: http://www.legislation.gov.uk/ukpga/1998/29/contents.

[53] US Congress, “Health Insurance Portability and Accountability Act of 1996 (HIPPA),” *104th United States Congress*, 1996. [Online]. Available: http://aspe.hhs.gov/admnsimp/pl104191.htm, 1996. [Accessed: 23-Apr-2013].

[54] D. Bowers, “What Will It Take to Get HIPPA-Compliant?,” *J. Heal. Care Compliance*, vol. 3, p. 48, 2001.

[55] Windows, “Windows Azure,” *Security data sheet*, 2011. [Online]. Available: http://www.microsoft.com/windowsazure/. [Accessed: 22-Jun-2011].

[56] Salesforce, “Salesforce website,” *Salesforce*. [Online]. Available: Available at: http://www.salesforce.com/. [Accessed: 28-Jul-2011].

[57] Salesforce, “Salesforce Security Datasheet,” 2010. [Online]. Available: http://www.salesforce.com/assets/pdf/datasheets/security.pdf.

[58] Marketo, “Marketo security,” *Marketo security*. [Online]. Available: http://www.marketo.com/security.php. [Accessed: 22-Jul-2011].

[59] Marketo, “Marketo.” [Online]. Available: http://www.marketo.com/about/. [Accessed: 22-Jul-2011].

[60] Zuora, “Zuora.” [Online]. Available: http://www.zuora.com/. [Accessed: 21-May-2012].

[61] Terremark, “Terremark Data sheet,” *Terremark Data Service*. [Online]. Available: http://www.terremark.com/uploadedFiles/Services/ADS/BackupDraftv2.pdf. [Accessed: 15-Jul-2011].

[62] A. Armando, R. Carbone, L. Campagna, J. Cuellar, and L. Tobarra, “Formal Analysis of SAML 2.0 Web Browser Single Sign-On: Breaking the SAML-Based Single Sign-On for Google Apps,” *6th ACM Work. Form. Methods Secur. Eng. FMSE*, 2008.

[63] E. Messmer, “Gartner: Cloud-based security as a service set to take off,” *Network World*, 2013. [Online]. Available: http://www.networkworld.com/article/2171424/data-breach/gartner--cloud-based-security-as-a-service-set-to-take-off.html. [Accessed: 21-Apr-2012].

[64] “Qualys.” [Online]. Available: www.qualys.com. [Accessed: 22-Jul-2012].

[65] C. Cloud, “Cipher Cloud.” [Online]. Available: http://www.ciphercloud.com/. [Accessed: 22-Aug-2012].

[66] D. Sheet, “Cipher Cloud for AWS,” *Cloud, Cipher*, 2014. [Online]. Available: http://www.ciphercloud.com/wp-content/uploads/2013/11/CipherCloud-for-Amazon-Web-Services.pdf. [Accessed: 22-Sep-2011].

[67] Qualys, “Qualys Guard.” [Online]. Available: https://www.qualys.com/solutions/compliance/cag/. [Accessed: 22-Aug-2012].

[68] G. Leopold, “Forecasts Call For Cloud Burst Through 2018,” *IDC*, 2014.

[69] X. Chen, G. B. Wills, L. Gilbert, and D. Bacigalupo, “Technical Report on Using Cloud for Research: A Technical Review,” University of Southampton, 2010.

[70] D. Catteddu and G. Hogben, “Cloud Computing: Benefits, risks and recommendations for information security,” ENISA, 2009.

[71] R. Boutaba, L. Cheng, and Q. Zhang, “Cloud computing: state-of-the-art and research challenges,” *Journal of Internet Services and Applications*, vol. 1, no. 1. pp. 7–18, 2010.

[72] B. R. Kandukuri, V. R. Paturi, and A. Rakshit, “Cloud Security Issues,” in *Services Computing, 2009. SCC ’09. IEEE International Conference on*, 2009, pp. 517–520.

[73] Z. S. Z. Shen and Q. T. Q. Tong, “The security of cloud computing system enabled by trusted computing technology,” *Signal Process. Syst. (ICSPS), 2010 2nd Int. Conf.*, vol. 2, 2010.

[74] K. Popović and Ž. Hocenski, “Cloud computing security issues and challenges,” *IEEE - Comput. Inf. Sci.*, vol. 3, no. 3, pp. 344–349, 2010.

[75] Department of Defense, “Trusted computer system evaluation criteria,” *{Department of Defense}*, pp. 1–116, 1985.

[76] B. Krebs, “Amazon: Hey Spammers, Get Off My Cloud!,” *The Washington Post*, 2008. [Online]. Available: http://voices.washingtonpost.com/securityfix/2008/07/amazon_hey_spammers_get_off_my.html.

[77] A. Ferdowsi, “Checksum Failure of Amazon S3,” 2008. [Online]. Available: https://forums.aws.amazon.com/thread.jspa?threadID=22709&start=0&tstart=. [Accessed: 12-Aug-2011].

[78] H. Lindqvist, “Mandatory access control,” UMEA University, 2006.

[79] D. F. Ferraiolo, D. M. Gilbert, and N. Lynch, “An examination of federal and commercial access control policy needs,” in *16th NIST-NSA National Computer Security Conference*, 1993.

[80] D. Ferraiolo, D. Kuhn, and R. Chandramouli, “Role-based access control,” in *15th National Computer Security Conference*, 2003, no. 1992, pp. 554 – 563.

[81] R. Sailer, T. Jaeger, E. Valdez, R. Caceres, R. Perez, S. Berger, J. L. Griffin, and L. van Doorn, “Building a MAC-based security architecture for the Xen open-source hypervisor,” *21st Annu. Comput. Secur. Appl. Conf.*, 2005.

[82] R. Sailer, E. Valdez, T. Jaeger, R. Perez, L. Van Doorn, J. L. Griffin, S. Berger, and Y. Heights, “IBM Research Report sHype : Secure Hypervisor Approach to Trusted Virtualized Systems sHype : Secure Hypervisor Approach to Trusted Virtualized Systems 1,” in *Science*, 2005, vol. 23511, no. RC23511.

[83] C. J. McCollum, J. R. Messing, and L. Notargiacomo, “Beyond the pale of MAC and DAC-defining new forms of access control,” *Proceedings. 1990 IEEE Comput. Soc. Symp. Res. Secur. Priv.*, 1990.

[84] R. Needham and R. MAYBURY, “Access Control,” in *Security Engineering: A guide to building dependable distributed systems*, University of Cambridge.

[85] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, *Role based access control models*, 29th ed. IEEE Computer, 1996.

[86] R. S. Sandhu and P. Samarati, *Access control: Principles and practice*, 32nd ed. IEEE Communication, 1994.

[87] W. Li, H. Wan, X. Ren, and S. Li, “A Refined RBAC Model for Cloud Computing,” in *Proceedings of the 2012 IEEE/ACIS 11th International Conference on Computer and Information Science*, 2012, pp. 43–48.

[88] P. Singh and S. Singh, “A New Advance Efficient RBAC to Enhance the Security in Cloud Computing,” *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 6, 2013.

[89] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, “Plutus: Scalable secure file sharing on untrusted storage,” in *FAST ’03: Proceedings of the 2nd USENIX Conference on File and Storage Technologies*, 2003, pp. 29 – 42.

[90] E. Goh, H. Shacham, N. Modadugu, and D. Boneh, “SiRiUS: Securing remote untrusted storage,” *Netw. Distrib. Syst. Secur. (NDSS ’03) Symp.*, no. 0121481, pp. 131–145, 2003.

[91] S. Hohenberger, K. Fu, M. Green, and G. Ateniese, “Improved proxy re-encryption schemes with applications to secure distributed storage,” *ACM Transactions on Information and System Security*, vol. 9, no. 1. pp. 1–30, 2006.

[92] S. Di Vimercati, S. Foresti, and S. Jajodia, “Over-encryption: management of access control evolution on outsourced data,” *Very large data bases*, pp. 123–134, 2007.

[93] M. Blaze, G. Bleumer, and M. Strauss, “Divertible protocols and atomic proxy cryptography,” *Adv. Cryptol. EUROCRYPT98*, vol. 1403, pp. 127–144, 1998.

[94] M. Blanton, N. Fazio, M. J. Atallah, and K. B. Frikken, “Dynamic and Efficient Key Management for Access Hierarchies,” *ACM Transactions on Information and System Security*, vol. 12, no. 3. pp. 1–43, 2009.

[95] S. Y. S. Yu, C. W. C. Wang, K. R. K. Ren, and W. L. W. Lou, “Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing,” *INFOCOM, 2010 Proc. IEEE*, 2010.

[96] A. Sahai and B. Waters, “Fuzzy Identity-Based Encryption,” *Eurocrypt*, vol. 3494, pp. 1–15, 2005.

[97] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” *Proc. 13th ACM Conf. Comput. Commun. Secur. CCS 06*, p. 89, 2006.

[98] M. Chase and S. S. M. Chow, “Improving privacy and security in multi-authority attribute-based encryption,” in *Proceedings of the 16th ACM conference on Computer and communications security - CCS ’09*, 2009, p. 121.

[99] L. Hu, S. Ying, X. Jia, and K. Zhao, “Towards an Approach of Semantic Access Control for Cloud Computing.,” in *CloudCom*, 2009, vol. 5931, pp. 145–156.

[100] GFI, “Control User Access to Endpoint Connections,” *GFI Endpoint Security*. [Online]. Available: http://www.gfi.com/landing/uk/esecstordev-uk.asp?adv=953&loc=4&gclid=CIaQ5tyus6UCFQkf4QodhlDnYQ. [Accessed: 12-Aug-2012].

[101] Symantec, “Symantec Data Loss prevention,” *Symantec*. [Online]. Available: http://www.symantec.com/en/uk/business/solutions/solutiondetail.jsp?solid=sol_info_risk_comp&solfid=sol_data_loss_prevention&om_sem_cid=biz_sem_emea_uk_Google_DLP. [Accessed: 12-Aug-2011].

[102] D. E. O’Leary, “Intrusion Detection Systems,” in *Journal of Information Systems (Spring)*, 1992, pp. 63–74.

[103] K. Wang and S. J. Stolfo, “Anomalous payload-based network intrusion detection,” *Recent Adv. Intrusion Detect.*, vol. 3224, pp. 203–222, 2004.

[104] D. Wagner and P. Soto, “Mimicry Attacks on Host-Based Intrusion Detection Systems,” in *Proceedings of the 9th ACM conference on Computer and communications security - CCS ’02*, 2002, p. 255.

[105] A. Ghosh, A. Schwartzbard, and M. Schatz, “Using program behavior profiles for intrusion detection,” in *… of the SANS Intrusion Detection …*, 1999, no. 703, pp. 1–7.

[106] K. Ilgun, “USTAT: a real-time intrusion detection system for UNIX,” *Proc. 1993 IEEE Comput. Soc. Symp. Res. Secur. Priv.*, 1993.

[107] T. D. Garvey and T. F. Lunt, “Model based intrusion detection,” *14th Natl. Comput. Secur. Conf.*, 1991.

[108] T. F. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, P. G. Neumann, and C. Jalali, “IDES: a progress report [Intrusion-Detection Expert System],” *[1990] Proc. Sixth Annu. Comput. Secur. Appl. Conf.*, 1990.

[109] T. F. Lunt, “A survey of intrusion detection techniques,” *Comput. Secur.*, vol. 12, no. 4, pp. 405–418, 1993.

[110] BBC, “Wikileaks: Suspect Bradley Manning faces 22 new charges,” *BBC News*, 2011. [Online]. Available: http://www.bbc.co.uk/news/world-us-canada-12628983. [Accessed: 12-Sep-2012].

[111] N. MacDonald, “Hypervisor Attacks in the Real World,” *Gartner*, 2009. [Online]. Available: http://blogs.gartner.com/neil_macdonald/2009/02/20/hypervisor-attacks-in-the-real-world/.

[112] E. Ray and E. Schultz, “Virtualization security,” *Proc. 5th Annu. Work. Cyber Secur. Inf. Intell. Res. Cyber Secur. Inf. Intell. Challenges Strateg. CSIIRW 09*, no. 1965, p. 1, 2009.

[113] Vmware, “Vmware nat networking buffer overflow vulnerability,” *Vm*. [Online]. Available: http://secunia.com/advisories/18162/. [Accessed: 12-Jun-2011].

[114] M. Carpenter, T. Liston, and E. Skoudis, “Hiding Virtualization from Attackers and Malware,” *IEEE Secur. Priv. Mag.*, vol. 5, no. 3, 2007.

[115] R. Naraine, “Blue Pill Prototype Creates 100% Undetectable Malware,” *eWEEK*, 2006. [Online]. Available: http://www.eweek.com/c/a/Windows/Blue-Pill-Prototype-Creates-100-Undetectable-Malware.

[116] S. T. King, P. M. Chen, Y. W. Chad, V. Helen, and J. W. Jacob, “SubVirt: Implementing malware with virtual machines,” *2006 IEEE Symp. Secur. Priv. SP06*, vol. 00, no. c, pp. 314–327, 2006.

[117] A. Khalid, “Testing Security of XEN using Dynamic and Static Analysis Techniques,” University of York.

[118] J. Wei, X. Zhang, G. Ammons, V. Bala, and P. Ning, “Managing security of virtual machine images in a cloud environment,” *Proc. 2009 ACM Work. Cloud Comput. Secur. - CCSW ’09*, no. Vm, pp. 91–96, 2009.

[119] P. A. Karger, “Multi-level security requirements for hypervisors,” *21st Annu. Comput. Secur. Appl. Conf.*, 2005.

[120] M. Christodorescu and S. Jha, “Static analysis of executables to detect malicious patterns,” in *SSYM’03 Proceedings of the 12th conference on USENIX Security Symposium*, 2003, vol. 12, pp. 12–12.

[121] M. Christodorescu, S. Jha, S. A. Seshia, D. Song, and R. E. Bryant, “Semantics-aware malware detection,” *2005 IEEE Symp. Secur. Priv.*, 2005.

[122] C. Kruegel, W. Robertson, and G. Vigna, “Detecting kernel-level rootkits through binary analysis,” *20th Annu. Comput. Secur. Appl. Conf.*, 2004.

[123] Checkmarx, “Checkmarx.” [Online]. Available: https://www.checkmarx.com. [Accessed: 20-Jun-2012].

[124] U. Bayer, A. Moser, C. Kruegel, and E. Kirda, “Dynamic Analysis of Malicious Code,” *EICAR*, 2006.

[125] Acunetix, “Acunetix.” [Online]. Available: https://www.acunetix.com/. [Accessed: 05-Apr-2015].

[126] J. Bergeron, M. Debbabi, J. Desharnais, M. M. Erhioui, Y. Lavoie, N. Tawbi, J. Bergeron, M. Debbabi, J. Desharnais, M. Erhioui, Y. Lavoie, and N. Tawbi, “Static Detection of Malicious Code in Executable Programs,” *Control*, pp. 184–189, 2001.

[127] Kali, “Kali Linux.” [Online]. Available: https://www.kali.org/. [Accessed: 05-Apr-2015].

[128] R. Goldberg, “Survey of virtual machine research,” *IEEE Computer Magazine*, pp. 34–45, Jun-1974.

[129] T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh, “Terra: A virtual machine-based platform for trusted computing,” in *Operating Systems Review (ACM)*, 2003, vol. 37, no. 5, pp. 193–206.

[130] T. Garfinkel and M. Rosenblum, “A Virtual Machine Introspection Based Architecture for Intrusion Detection,” in *Proc. Network and Distributed Systems Security …*, 2003, vol. 1, pp. 253–285.

[131] VMware, “Vmware Inc.” [Online]. Available: www.vmware.com. [Accessed: 12-Jun-2011].

[132] L. Litty, “Hypervisor based intrusion detection.,” University of Toronto, 2005.

[133] M. Dou El Kefel and B. Mohamed, “Risk management in cloud computing,” in *Innovative Computing Technology (INTECH), 2013 Third International Conference on*, 2013, pp. 127–131.

[134] S. Tanimoto, C. Murai, Y. Seki, M. Iwashita, S. Matsui, H. Sato, and A. Kanai, “A Study of Risk Management in Hybrid Cloud Configuration,” in *Computer and Information Science SE - 18*, vol. 493, R. Lee, Ed. Springer International Publishing, 2013, pp. 247–257.

[135] CSA, “CSA Guidance,” 2012. [Online]. Available: https://cloudsecurityalliance.org/csaguide.pdf.

[136] A. Shamir, “How to share a secret,” *Communications of the ACM*, vol. 22, no. 11. pp. 612–613, 1979.

[137] D. Brink, “Security and Cloud Best Practices,” *Aberdeen Gr.*, 2011.

[138] A. U. Khan, M. Oriol, M. Kiran, M. Jiang, and K. Djemame, “Security risks and their management in cloud computing,” in *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, 2012, pp. 121–128.

[139] IRAM, “Information risk analysis methodology,” 2012. [Online]. Available: https://www.securityforum.org/iram#iramtva. [Accessed: 09-Nov-2012].

[140] P. Mcnamaara, “DDoS attack against Bitbucket darkens Amazon cloud,” *Network World*, 2009. [Online]. Available: http://www.networkworld.com/community/node/45891. [Accessed: 05-Aug-2010].

[141] ISO, “ISO 27002 Standard,.” [Online]. Available: http://www.27000.org/iso-27002.htm. [Accessed: 09-Nov-2012].

[142] J. M. Grusha N., “Attack Surfaces: A Taxonomy for Attacks on Cloud services,” in *IEEE 3rd Int Conference on Cloud Computing*, 2010.

[143] Google, “Security Whitepaper: Google Apps Messaging and Collaboration Products,” 2010. [Online]. Available: https://docs.google.com/fileview?id=0B5Y-fwYJF2hLY2MwNzk0NjQtYTZlNy00MjdiLThmMmUtOTM1OTRjMTdjNDMx&hl=en.

[144] Wikileaks, “Wikileaks.org.” [Online]. Available: http://wikileaks.org/. [Accessed: 29-Oct-2012].

[145] J. Garmon, “Endusers:The achilles heel in googleapps,” *Backupify*, 2010. [Online]. Available: http://www.backupify.com/resources/end_users.pdf?&utm_source=resources&utm_medium=whitepaper&utm_campaign=EndUserAchillesHeel. [Accessed: 01-Jan-2012].

[146] T. Shifrin, “Human error cost most data loss,studysays,” *PCWorld*, 2007. [Online]. Available:

http://www.pcworld.com/article/129736/human_error_causes_most_data_loss_study_says.html. [Accessed: 02-May-2012].

[147] P. Gordon, “Data Leakage-Threat and Mitigation,” *SANS Institute*, 2007. [Online]. Available: http://www.sans.org/reading_room/whitepapers/awareness/data-leakage-threats-mitigation_1931. [Accessed: 12-Sep-2012].

[148] I. Thomson, “Google Docs Leaks Private User Data Online,” *v3*, 2009. [Online]. Available: http://www.v3.co.uk/vnunet/news/2238122/google-docs-leaks-private.

[149] Blackhat, “Cloudburst: Arbitrary code execution vulnerability for VMWare,” 2009. [Online]. Available: http://www.blackhat.com/presentations/bh-usa-09/KORTCHINSKY/BHUSA09-Kortchinsky-Cloudburst-SLIDES.pdf. [Accessed: 12-Sep-2012].

[150] T. Ormandy, “An Empirical Study into the Security Exposure to Hosts of Hostile Virtualized Environments,” *Test*, pp. 1–10, 2007.

[151] S. Yilek, “Resettable Public-Key Encryption : How to Encrypt on a Virtual Machine,” in *CT-RSA 2010: Topics in Cryptology*, 2010, pp. 41–56.

[152] W. J. Clinton and A. Gore, “A Framework for Global Electronic Commerce,” *IT Law Wiki*. [Online]. Available: http://itlaw.wikia.com/wiki/A_Framework_for_Global_Electronic_Commerce. [Accessed: 21-Jun-2011].

[153] R. R. Schriver, “You Cheated, You Lied: The Safe Harbor Agreement and Its Enforcement,” *Fordham Law Review*, 2002. [Online]. Available: http://ir.lawnet.fordham.edu/cgi/viewcontent.cgi?article=3848&context=flr. [Accessed: 12-Aug-2011].

[154] BitBucket, “Another EBS/EC2 incident,” *BB*, 2010. [Online]. Available: http://blog.bitbucket.org/2010/04/20/another-ec2ebs-incident-what-happened/. [Accessed: 23-Jan-2013].

[155] F. Y. Rashid, “Wikileaks no longer on Amazon EC2, return to Swedish hosting service,” *eWEEK*. [Online]. Available: http://www.eweek.com/c/a/Security/WikiLeaks-No-Longer-on-Amazon-EC2-Returns-to-Swedish-Hosting-Service-671260/. [Accessed: 14-Nov-2012].

[156] A. Ferdowsi, “Checksum Failure of Amazon S3,” *Amazon*. [Online]. Available: https://forums.aws.amazon.com/thread.jspa?threadID=22709&start=0&tstart=. [Accessed: 12-Aug-2011].

[157] A. U. Khan and T. Dimitrakos, “Keys for protecting user access to media,” PCT/GB2009/000793, 2009.

[158] A. Perrig, R. Canetti, J. Tygar, and D. Song, “The TESLA broadcast authentication protocol,” *CryptoBytes*, vol. Summer/Fal, no. 5:2, pp. 2–13, 2002.

[159] S. Wee and J. Apostolopoulos, “Secure scalable video streaming for wireless networks,” *IEEE Int. Conf.*, 2001.

[160] S. Wee and J. Apostolopoulos, “Secure scalable streaming and secure transcoding with JPEG-2000,” *Proc. 2003 Int. Conf. Image Process. (Cat. No.03CH37429)*, vol. 1, 2003.

[161] S. J. Wee and J. G. Apostolopoulos, “Secure scalable streaming enabling transcoding without decryption,” *Proc. 2001 Int. Conf. Image Process. (Cat. No.01CH37205)*, vol. 1, 2001.

[162] G. R. Blakley, “Safeguarding cryptographic keys,” in *AFIPS*, 1979, p. 313.

[163] QMU, “Multimedia & Vision Research Group,” *Queen Mary University*, 2006. [Online]. Available: http://www.eecs.qmul.ac.uk/group/mmv/svc.html. [Accessed: 07-Sep-2012].

[164] Secure Cloud Storage, “Huddle,” *Huddle*, 2013. .

[165] K. Scarfone and A. Orebaugh, “Technical Guide to Information Security Testing and Assessment Recommendations of the National Institute of Standards and Technology,” *Nist Spec. Publ.*, vol. 800, pp. 1–80, 2008.

[166] F. Whiteside, L. Badger, M. Iorga, J. Mao, and S. Chu, “Challenging Security Requirements for US government Cloud Computing Adoption,” *Spec. Publ. 500-296*, 2013.

[167] M. Kiran, A. U. Khan, M. Jiang, K. Djeame, M. Oriol, and M. Corrales, “Managing Security Threats in Clouds,” *Digit. Res.*, 2012.

[168] A. U. Khan, M. Oriol, and M. Kiran, “Threat Methodology for Securing Scalable Video in the Cloud,” *8th IEEE Int. Conf. Internet Technol. Secur. Trans. Proceedings, 2013*, 2013.

[169] A. U. Khan, F. La Torre, and M. Oriol, “Network and Access Security System,” Patent Application Number: 13715392, 2013.

[170] A. U. Khan, F. La Torre, and M. Oriol, “METHOD AND SYSTEM FOR NETWORK DATA ACCESS,” PCT/GB2013/000145, 2013.

[171] L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker, “Mediated Ciphertext-Policy Attribute Based Encryption and its Application,” *Work. Inf. Secur. Apl. (WISA ‘09)*, pp. 309–323, 2009.

[172] S. Yu, C. Wang, and W. Lou, “Attribute Based Data Sharing with Attribute Revocation,” in *Proceeding ASIACCS ’10 Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, 2010, pp. 261–270.

[173] Edney and W. A. Arbaugh, *Real 802.11 Security: Wi-Fi Protected Access and 802.11I*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2003.

[174] T. Hardjono and L. R. Dondeti, *Security In Wireless LANS And MANS (Artech House Computer Security)*. Norwood, MA, USA: Artech House, Inc., 2005.

[175] Oasis, “Oasis XACML.” [Online]. Available: http://www.oasis-open.org/committees/xacml/. [Accessed: 14-Nov-2012].

[176] D. Nunez, “NICS Crypto Library,” *NICS*, 2012. [Online]. Available: https://www.nics.uma.es/dnunez/nics-crypto. [Accessed: 24-Jun-2014].

[177] D. Nunez, I. Agudo, and J. Lopez, “Integrating OpenID with proxy re-encryption to enhance privacy in cloud-based identity services,” in *CloudCom 2012 - Proceedings: 2012 4th IEEE International Conference on Cloud Computing Technology and Science*, 2012, pp. 241–248.