Ensuring File Security on Cloud using Two-tier Encryption and Decryption

Thesis submitted in partial fulfillment of the requirements for the award of degree

of

Masters of Technology

in

Computer Science and Applications

Submitted By
Tanisha
(Roll No. 601103025)

Under the supervision of
Dr. Rajesh Kumar
Associate Professor, SMCA

School of Mathematics and Computer Applications
Thapar University
Patiala –147004

July 2013

Certificate

I hereby certify that the work which is presented in the thesis entitled, " **Ensuring File Security on Cloud using Two-tier Encryption and Decryption"**, in partial fulfillment of the requirements for the award of degree of the Master of Technology in **Computer Science and Applications,** submitted in School of mathematics and Computer Applications of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of **Dr. Rajesh Kumar** and refers others researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other university.

Tanisha

Tanisha

601103025

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

27/06/2013

Dr. Rajesh Kumar

Associate Professor

School of Mathematics and Computer Applications

Countersigned by

Dr. Rajesh Kumar

Head

School of Mathematics and Computer Applications

Thapar University

Patiala

Dr. S.K. Mohapatra

Dean Academic Affairs

Thapar University

Patiala

i

Acknowledgement

I would like to acknowledge everyone who supported me during my experience at Thapar University and my work on this thesis.

First of all, I would like to thank my family, who stayed with me and supported me all the time while I made this dream come true. I extend my thanks to my thesis supervisor, Dr. Rajesh Kumar, who helped me to materialize my ideas on this thesis. His enthusiasm and optimism made this experience both rewarding and enjoyable. Most of the novel ideas and solutions found in this thesis are the result of our numerous stimulating discussions. His feedback and editorial comments were also valuable for writing this thesis.

My thesis work would be incomplete without thanking my friends who were always there in the hour of need.

Abstract

Cloud computing is a model that offers incredible processing power and on-demand network access to a shared pool of configurable computing resources like networks, servers, storage, applications, and services. In the new era of the modern world, cloud computing has become a popular choice among the corporate structures and individual consumers. Information technology is undergoing a radical change because of cloud computing. With the increasing popularity of the cloud technology, security of information becomes much important in data storage and transmission**.** The cloud can't guarantee absolute data security, thus, there is a need to take safety measures for data protection at one's own end. It is important to ensure that the stored data is neither compromised nor corrupted, thus making authentication and authorization for data access, a necessity.

The proposed methodology suggests a new security scheme for the files to be uploaded on the cloud. The integrity and confidentiality is ensured by encrypting the data using a combination of two-tier hybrid encryption and the digital signature scheme and also, by providing access to the data only on successful authentication.

List of Figures

List of Tables

Table of Contents

Chapter 1

Introduction

1.1 Overview of Cloud Computing

Cloud computing is an emerging area within the field of Information Technology. It has brought a revolution in IT industry by enabling the use of storage, processing, or higher level elements such as operating systems or software applications, not by owning them and having them installed on computers but rather to use these resources simply as a service. Cloud computing deals with computation, software, data access and storage services that may not require end-user knowledge of the physical location and the configuration of the system that is delivering the services. It is a recent trend in IT that moves computing and data away from desktop and portable PCs into large data centers. From a generic point of view, it could be said that cloud computing is a kind of computing where scalable, adaptable, and elastic IT capabilities are provided as a service to multiple users.

As per 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".

Cloud computing has been coined as an umbrella term to describe a category of sophisticated on- demand computing services initially offered by commercial providers, such as Amazon, Google, and Microsoft. It denotes a model on which a computing infrastructure is viewed as a "cloud", from which businesses and individuals access applications from anywhere in the world on demand [1]. The main goal of cloud computing is to make a better use of distributed resources, combine them to achieve higher throughput and be able to solve large scale computation problems.

1.2 Cloud Computing Deployment Models

Based upon the underlying infrastructure, clouds can be classified as Public, Private, Hybrid and Community clouds. The different infrastructure deployment models are distinguished by their architecture, the location of the datacenter where the cloud is realized, and the needs of the cloud provider's customers.

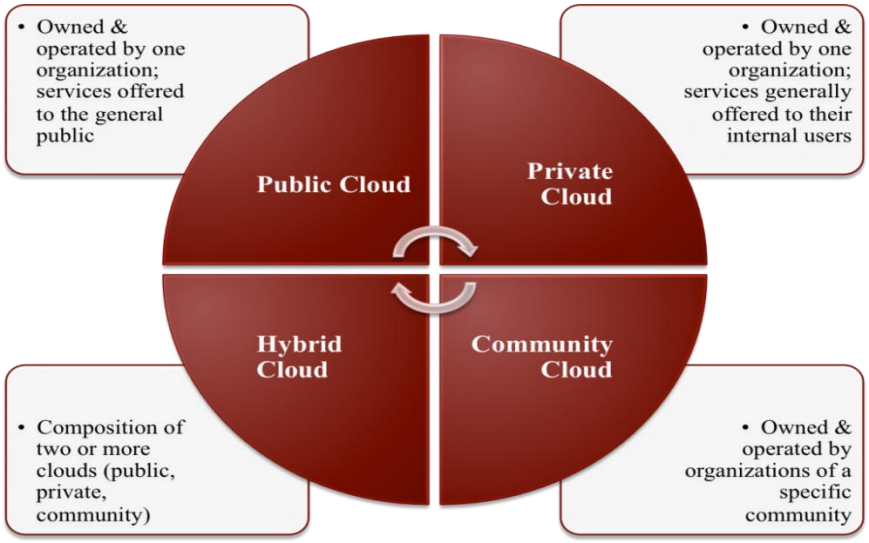

Figure 1.1: Deployment Models [2]

1.2.1 Public Cloud

Public cloud is defined as a "cloud made available to the general public in a pay as you go manner" [3]. A public cloud's physical infrastructure is owned by a cloud service provider. In a public cloud, resources are shared with multiple customers, which may operate in different market segments, and may have different security demands.

1.2.2 Private Cloud

Private cloud is defined as "internal data center of a business or other organization, not made available to the general public [3]."

A pure private cloud is built for the exclusive use of an organization, which owns and fully controls this cloud. Compared to public cloud where all the resources and applications are managed by the service provider, in private cloud these services are pooled together and made available for the users at the organizational level.

1.2.3 Community Cloud

When several customers have similar requirements, they can share an infrastructure and might share the configuration and management of the cloud. This management might be done by themselves or by third parties.

A community cloud is defined as a "cloud infrastructure shared by the several organizations and supports a specific community that has shared concerns e.g., mission, security requirements, policy, and compliance considerations" [3].

1.2.4 Hybrid Cloud

A hybrid cloud is defined as "a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability"[4].

The hybrid cloud phenomenon helps the organizations to achieve maximum usability, while minimizing the degree of faults. Any composition of clouds, be they private or public, could form a hybrid cloud and be managed a single entity, provided that there is sufficient commonality between the standards used by the constituent clouds.

1.3 Cloud Computing Service Models

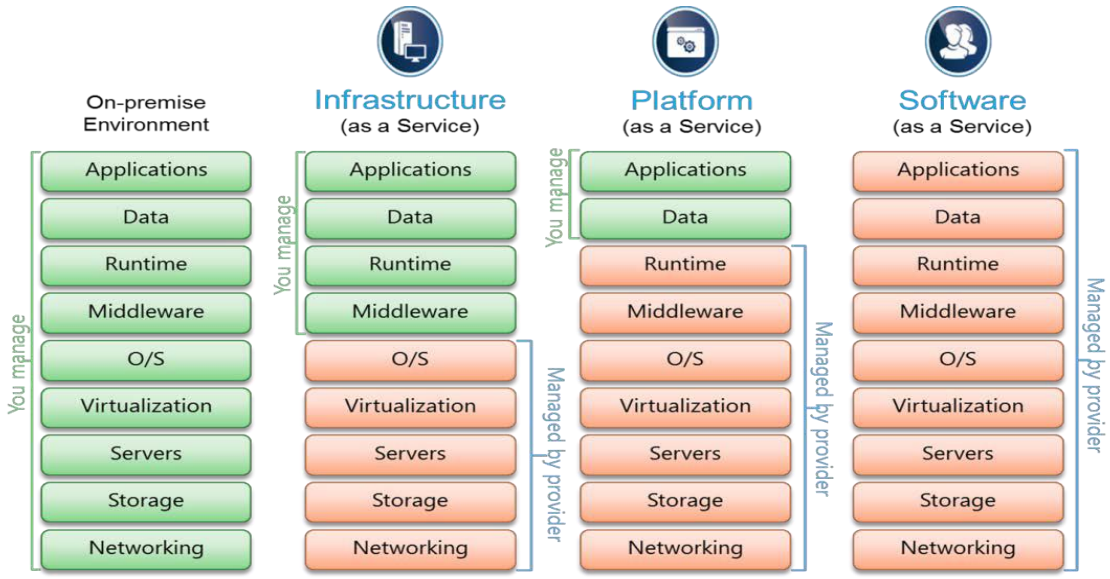

Figure 1.2: Service Models [5]

1.3.1 Infrastructure as a Service (IaaS)

Infrastructure as a Service is a provision model in which an organization outsources the equipment used to support operations, including storage, hardware, servers and networking components [6]. The service provider owns the equipment and is responsible for housing, running and maintaining it. The client typically pays on a per-use basis.

As per NIST,

"IaaS is a capability provided to the consumer to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The cloud customer does not manage or change parameters of the underlying infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls)."

1.3.2 Platform as a Service (PaaS)

Platform as a Service is a delivery model that allows the customers to rent virtualized servers and associated services for running existing applications or developing and testing new ones [6]. In case of PaaS, the cloud provider not only provides the hardware, but they also provide a toolkit and a number of supported programming languages to build higher level services.

As per NIST,

"PaaS is a capability provided to customer to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations".

1.3.3 Software as a Service (SaaS)

Software as a Service (SaaS) is software distribution model in which applications are hosted by a vendor or service provider and made available to customers over a network [6].

As per NIST,

"SaaS is a capability to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through a thin client interface such as a web browser. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings".

1.4 Features of Cloud

i) *On Demand Service:* Cloud computing is based on self-service and on-demand service models. It allows the user to interact with the cloud to perform tasks like building, deploying, managing, and scheduling. The user should be able to access computing capabilities as and when they are needed and without any interaction from the cloud-service provider.

ii) *Pay Per-Usage:* Cloud computing does not have any upfront cost. It is completely based on resource usage such as processing cycles, disk space etc. Cloud computing services use a metering capability which enables to control and optimize resource use. This implies that just like other utility services e.g. electricity, IT services are charged on per usage metrics – pay per use. The more you utilize the higher you pay.

iii) *Resource Pooling:* Cloud computing uses multi-tenancy where different resources are dynamically allocated and de-allocated according to demand. In a multi-tenant cloud, there exists a great disparity between users. Thus, resources rented from the cloud must be highly customizable in order to serve a large number of customers. The resources include storage, processing, memory, network bandwidth, virtual machines.

iv) *Elasticity:* Cloud computing gives the illusion of infinite computing resources available on demand. Therefore users expect clouds to rapidly provide resources in any quantity at any time. The resource allocation should be elastic, in the sense that it should change appropriately and quickly with the demand. If on a particular day the demand increases several times, then the system should be elastic enough to meet that additional need (scale up), and should return to the normal level (scale down) when the demand decreases.

1.5 Open Source Cloud Computing Frameworks

Eucalyptus, OpenNebula and Nimbus are three major open-source cloud-computing software platforms. The overall function of these systems is to manage the provisioning of virtual machines for a cloud providing infrastructure-as-a-service. These various open-source projects provide an important alternative for those who do not wish to use a commercially provided cloud.

i) *Eucalyptus:* This is an open source cloud computing framework developed by the University of California at Santa Barbara as an alternative to Amazon EC2. The Eucalyptus framework was one of the first open-source projects to focus on building IaaS clouds. It has been developed with the intent of providing an open-source implementation nearly identical in functionality to Amazon Web Services APIs. Therefore, users can interact with a Eucalyptus cloud using the same tools they use to access Amazon EC2 [7]. It also distinguishes itself from other tools because it provides a storage cloud API—emulating the Amazon S3 API—for storing general user data and VM images.

ii) *Nimbus:* The Nimbus toolkit is built on top of the Globus framework. Nimbus provides most common features available with other open-source VI managers, such as an EC2-compatible front-end API, support to Xen, and a backend interface to Amazon EC2 [8]. The Nimbus platform is targeted to provide additional tools to simplify the management of infrastructure services and to facilitate the integration with other existing clouds. Nimbus is incredibly customizable. Nimbus' core was engineered around the Spring framework to be easily extensible, thus allowing several internal components to be replaced and also eases the integration with other systems.

iii) *OpenNebula:* Open Nebula is an open source IaaS toolkit. The idea of OpenNebula is a pure private cloud, in which users actually log into the head node to access cloud functions [7]. Its design is flexible and modular to allow the integration with different storage and network infrastructure configurations and hypervisor technologies. It can deal with changing resource needs, resource additions, live migration etc. It supports the cloud federation to interface with external clouds which provides scalability.

OpenNebula tends to a greater level of centralization and customizability especially for its end users.

iv) *OpenStack:* OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter [9]. It is an open source IaaS for building and managing public and private clouds. The goals of the OpenStack initiative are to support interoperability between cloud services and allow businesses to build Amazon-like cloud services in their own data centers. OpenStack has a modular architecture that currently has three components. These components include OpenStack Compute (Nova), OpenStack Object Storage (Swift) and OpenStack Image Service (Glance).

1.6 Benefits of Cloud Computing

i) *Lower Capital Expenditure:* The ability to source IT services on demand as and when they are required allows businesses to move to an investment model based on operational expenditure. Businesses simply become consumers of IT services rather than the owners of the hardware. Organizations are no longer required to spend thousands on software licenses which may only have a limited lifespan. Also, they can save the considerable costs associated with building, maintaining, and operating a data center, especially power and cooling related expenditures. When businesses source IT services in the cloud, it is the vendor that takes responsibility for the majority of the infrastructure.

ii) *Scalability:* Enterprises no longer need to invest time in buying and setting up the hardware, software and other resources necessary for a new application. Cloud providers offer a burstable infrastructure that allows businesses to quickly scale up or scale down their usage of services on the cloud as per market demands, thereby allowing flexibility as the needs change.

iii) *Optimum Resource Utilization:* The advent of virtualization has provided companies with ways to efficiently use their computer resources i.e. servers, storage and network resources. Virtualization enables multiple server technologies to run from a single server. Thus, users no longer require separate servers for different

applications. This shift to virtualization supports the growth of cloud computing by increasing the capabilities of servers.

iv) *Location Independence:* A cloud computing service can be used from anywhere, unlike the most physical infrastructures that are tied down to one particular place. Instead of running applications and storing data on the personally owned hardware, the applications run on a shared data center. There is no limitation of place and medium. People worldwide can access the cloud, anytime; provided they have an internet connection.

v) *Backup and Recovery:* The process of backing up and recovering data is simplified since; data now reside on the cloud and not on any physical device. The various cloud providers offer reliable and flexible backup/recovery solutions. In some cases, the cloud itself is used solely as a backup repository of the data located in local computers.

vi) *Environmentally friendly:* The cloud is more efficient than the typical IT infrastructure as it takes fewer resources to compute, thus saving energy. For example, when servers are not in use, the infrastructure normally scales down, freeing up resources and consuming less power. At any moment, only the resources that are truly needed are consumed by the system.

1.7 Security Issues in Cloud

Cloud computing is the new era of the modern world. IT sector is rapidly moving onto cloud as the organizations can now use the best resources in a cost effective way. As more and more information is moved to the cloud, security concerns have started to develop. Though benefits of the cloud are enormous but cloud has got many issues when it comes to security. Some of the security issues related to the cloud computing are described as below [10, 11, 12]:-

i) *Data Integrity:* Data stored in the cloud typically resides in the shared environment. When a data is on a cloud, anyone from any location can access those data and information from the cloud. Cloud does not differentiate between a sensitive data from a common data thus enabling anyone to access the sensitive data. The cloud computing service provider must make sure that the customer‘s private and sensitive

data is well secured from other providers and users. The service provider should maintain control over access by imposing several access control rights and privileges to keep a check on who is accessing the data and who is maintaining the server. Besides access control, authentication is also required so that only a legitimate user can access the data not any malicious user.

ii) *Data Stealing:* Data stealing is a one of serious issue in a cloud computing environment. Data stealing refers to the illegal acquisition of information. Many cloud service providers do not provide their own server instead they lease server from other service providers due to its cost effectiveness and flexibility. So there is a high probability that data can be stolen from the external server.

iii) *Data Loss:* Data loss is a common problem in cloud computing. A malicious hacker can wipe out the data or any natural/man-made disaster can destroy or corrupt the data leading to the loss of data. In such cases, having an offline copy is of great importance. Moreover, even the carelessness of the service provider can also lead to data loss e.g. if the service provider shut down his services due to some financial or legal problem then there will be a loss of data for the user.

iv) *Infected Application:* In the cloud computing environment**,** there exists a possibility where a malicious user can penetrate the cloud by acting as a legitimate user, there by infecting the entire cloud and thus affecting many customers who are sharing the infected cloud. It is the responsibility of the service provider to prevent any malicious user from uploading any infected application onto the cloud. Cloud computing service provider should have the complete access to the server with all rights for the purpose of monitoring and maintenance of server.

v) *User Level Security:* Even though the service provider takes necessary measures to provide a good security layer for the customer , it is also the responsibility of the customer to make sure that because of its own action, there shouldn’t be any loss of data or tampering of data . The customer can take essential measures to protect the privacy of the sensitive data to be stored on the remote servers e.g. encrypting the documents before uploading to remote servers is one such measure that can be adopted.

1.8 Need for Data Security in Clouds

Cloud computing is one of new trends in the field of information technology. Social media channels, corporate structures and individual consumers are all switching to the magnificent world of cloud computing. However, cloud computing faces many critical issues. Bear the brunt of them is data security, which has become an important factor restricting the development of cloud computing. Cloud computing stores data in the remote servers that are not owned or operated by users, thus data is easily drifted away from the user's control leading to various data security issues [13]. Whenever a data is on the cloud, anyone from anywhere can access data from the cloud anytime, hence making the data or files more vulnerable to attack. As a result, it is very easy for an intruder to access, misuse and destroy the original form of data. Although cloud architecture has its own security, but there exists an environment where the remote server can't be trusted and it is necessary to assure that the data has not been tampered with. Hence, it is extremely essential for the cloud to be secure [14].

Clouds typically have single security architecture but have many customers with different demands. In case of compromise by service provider at any cost, entrusting cloud side is of no use. Also at the same time, data must not be tampered by cloud server. Integrity is the core process to ensure storage security in cloud. Hence, it is necessary to take measures at the user's end to ensure security and integrity.

1.9 Key Components of Data Security

There are four key components of data security:

i) *Availability:* Data availability ensures continuous access to data even in the event of a natural or man-made disaster or events such as fires or power outages.

ii) *Integrity:* Data integrity ensures that the data is maintained in its original state and has not been intentionally or accidentally altered.

iii) *Confidentiality:* Data confidentiality means information is available or disclosed only to authorize individuals, entities, or IT processes.

iv) *Traceability:* Data traceability means that the data, transactions, communications, or documents are genuine and that both parties involved are who they claim to be.

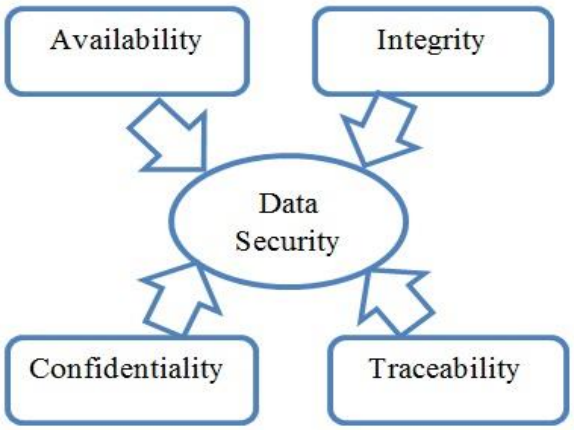

Figure 1.3: Components of Data Security

1.10 Cryptography

"*Cryptography* is the art and science of designing or generating the secret message i.e. code or ciphers of the original message for the secure communication between sender and the receiver." [15]

A cryptographic algorithm is a set of mathematical functions and unchanging set of steps to perform encryption and decryption of the original data.

Usually, two related functions are used, one for encryption and the other for decryption. With most modern cryptography, the ability to keep encrypted information secret is based not on the cryptographic algorithm, which is widely known, but on a number called a key that must be used with the algorithm to produce an encrypted result or to decrypt previously encrypted information.

The main objective of every cryptographic algorithm is to make it as difficult as possible to decrypt the generated cipher text without using the key.

1.10.1 Encryption

Process to transform or convert the data into some another form that appears to be random, meaningless and unintelligible. This is usually accomplished using a secret encryption key and a cryptographic cipher. It can also be said that encryption is the process of transforming plaintext into the cipher text where plaintext is the input to the encryption process and cipher text is the output of the encryption process. It is considered

as the subset of cryptography. Encryption is the most effective way to achieve data security.

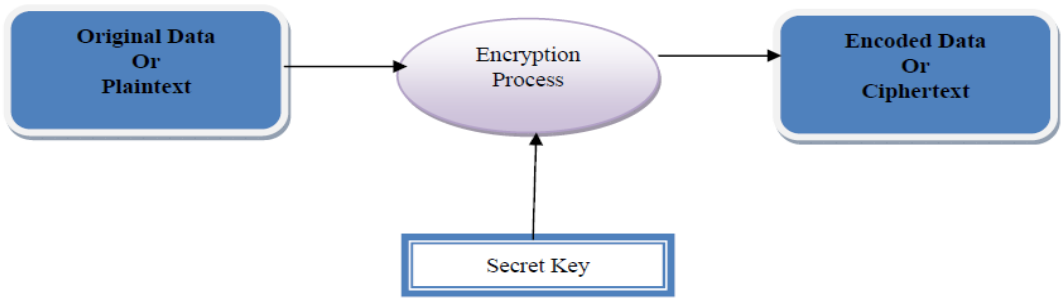

Figure 1.4: Encryption Process [15]

1.10.2 Decryption

Process to transform or convert the encoded data into some meaningful form. It can also be said that decryption is the process of transforming cipher text into the plaintext where cipher text is the input to the decryption process and plaintext is the output of the decryption process. Decryption without the correct key is very difficult, if not impossible.

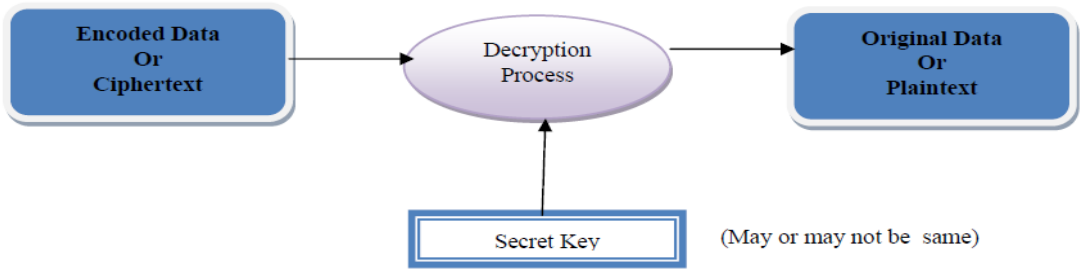

Figure 1.5: Decryption Process [15]

1.11 Goals of Cryptography

The main goals of cryptography are:

i) *Confidentiality:* It is the process of keeping information secret from all, but those who are authorized to see it. Confidentially is the protection of transmitted data from passive attacks. With respect to the content of data transmission, several levels of protection can be identified. The broadest service protects all user data

transmitted between two users over a period of time. The aspect of confidentially is the protection of traffic flow from analysis. This requires that an attacker should not be able to observe source and destination, frequency, length or any other characteristics of the traffic on a communication network.

ii) *Integrity:* It ensures that the information has not been altered by unauthorized or unknown means. One must have the ability to detect data manipulation by unauthorized parties. Data manipulation includes such things as insertion, deletion, and substitution.

iii) *Authentication:* It is a service related to identification. Data Authentication implicitly provides data integrity and it is applied to both entities and information itself, because any two parties entering into a communication should identify each other. Information delivered over a channel should be authenticated as to data origin, data content.

iv) *Non-repudiation:* Non-repudiation prevents either sender or receiver from denying a message. Thus, when a message is sent, the receiver can prove that the message was sent by the alleged sender. Similarly, when a message is received, the sender can prove that the alleged receiver received that message.

1.12 Cryptographic Techniques

There are two types of cryptographic techniques:

i) Symmetric Key Encryption

ii) Asymmetric Key Encryption

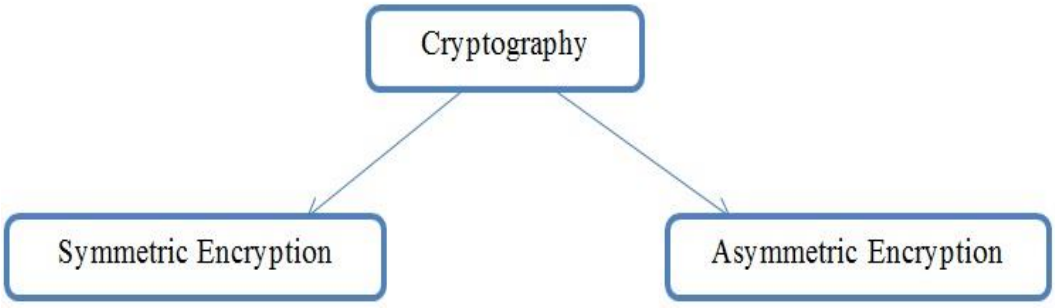

Figure 1.6: Types of Cryptographic Algorithms

1.12.1 Symmetric Key Encryption

Symmetric Key Encryption uses only one key for both encryption and decryption process. The key is transmitted to both the sender and receiver before the process of encryption and decryption. In the symmetric-key encryption, the encryption key can be calculated from the decryption key and vice versa. The secret key plays an important role and its strength depends on the length of key.

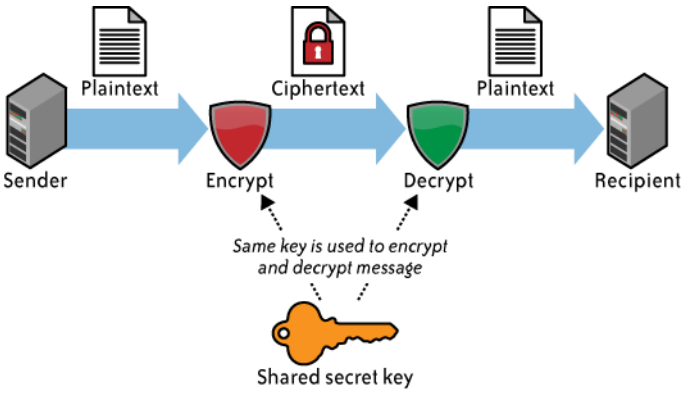

Figure 1.7: Symmetric Key Encryption [16]

Implementation of symmetric-key encryption can be highly efficient, so that users do not experience any significant time delay as a result of the encryption and decryption process. Symmetric-key encryption also provides a degree of authentication, since information encrypted with one symmetric key cannot be decrypted with any other symmetric key. Thus, as long as the symmetric key is kept secret by the two parties using it to encrypt communications, each party can be sure that it is communicating with the other as long as the decrypted messages continue to make sense.

1.12.2 Asymmetric Key Encryption

Asymmetric-key encryption (also called public-key encryption) involves a pair of keys, a public key and a private key, associated with an entity. Each public key is published, and the corresponding private key is kept secret. Data encrypted with a public key can be decrypted only with the corresponding private key.

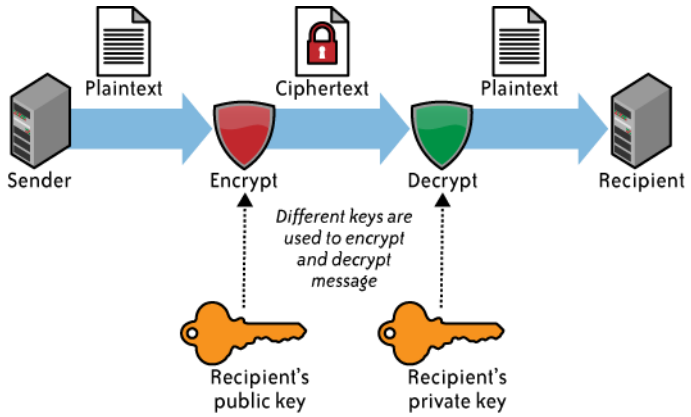

Figure 1.8: Asymmetric Key Encryption [16]

This cryptographic scheme allows public keys to be freely distributed, while only authorized people are able to read encrypted data using their corresponding private keys. In general, to send encrypted data, the data is encrypted with that person's public key, and the person receiving the encrypted data decrypts it with the corresponding private key.

Compared with symmetric-key encryption, public-key encryption requires more processing and may not be feasible for encrypting and decrypting large amounts of data. However, it is possible to use public-key encryption to send a symmetric key, which can then be used to encrypt additional data. This approach is used by the SSL/TLS protocols.

1.13 Hybrid Cryptosystem

Symmetric ciphers are significantly faster than asymmetric ciphers, but require the communicating parties to somehow share a secret key. Asymmetric ciphers have the advantage of increased security and convenience over the symmetric ciphers. Asymmetric cryptosystems are convenient in the sense that they do not require the sender and receiver to share a common secret key in order to communicate securely. However, they often rely on complicated mathematical computations and are thus generally much more inefficient than comparable symmetric-key cryptosystems. The asymmetric algorithms provide a high end security, but at the cost of speed. In many applications, the high cost of encrypting long messages in a public-key cryptosystem can be prohibitive.

A hybrid cryptosystem is a protocol using multiple ciphers of different types together, each to its best advantage i.e. the one that combines the convenience of a public-key cryptosystem with the efficiency of a symmetric-key cryptosystem [17].

To encrypt a message in a hybrid cryptosystem, sender does the following:

i) Obtains receiver's public key.
ii) Generates a fresh symmetric key for the data encapsulation scheme.
iii) Encrypts the message under the data encapsulation scheme, using the symmetric key.
iv) Encrypts the symmetric key, using receiver's public key.
v) Sends both of these encryptions to receiver.

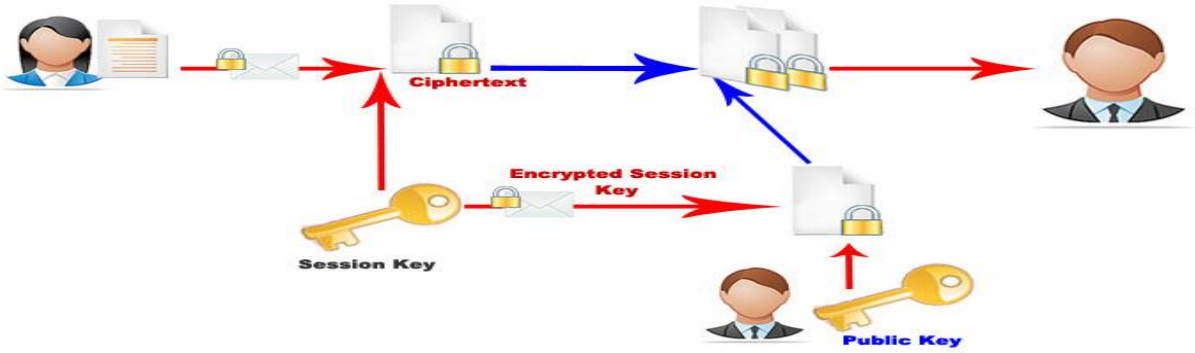

Figure 1.9: Hybrid Encryption Scheme [17]

To decrypt hybrid cipher text, receiver does the following:

i) Uses her private key to decrypt the symmetric key contained in the key encapsulation segment.
ii) Uses this symmetric key to decrypt the message contained in the data encapsulation segment.

Figure 1.10: Hybrid Decryption Scheme [17]

1.14 Key Management in Cryptography

The two components required to encrypt data includes an algorithm and a key. The algorithm is generally known, and the key is kept secret. The key is a very large number that should be impossible to guess, and of a size that makes exhaustive search impractical. A major issue in using the different cryptographic techniques is the management of the keys. Key Management deals with the secure generation, distribution and storage of keys. Secure methods of key management are extremely important, since most of the attacks are aimed at key management level, rather than at algorithm itself.

As the entire operation is dependent upon the security of the keys, it is sometimes appropriate to devise a fairly complex mechanism to manage them. When a single individual is involved, often direct input of a value or string will suffice. The 'memorized' value will then be re-input to retrieve the data, similar to password usage. Sometimes, many individuals are involved, with a requirement for unique keys to be sent to each for retrieval/decryption of transmitted data. In this case, the keys themselves may be encrypted [18]. A number of comprehensive and proven key management systems are available for these situations.

Chapter 2

Literature Survey

In this chapter we have surveyed various cryptographic schemes and data security issues in cloud environment and mapped them to our taxonomy to guide future design and development efforts.

2.1 Taxonomy of Cloud Computing System

B. Rimal *et al.* [4] have surveyed various cloud computing environments and services developed by various projects such as Google, force.com, amazon, open source. The surveyed results are used to identify differences, similarities and differences of the architectural approaches of cloud computing. The author defines the taxonomy and comparative study of cloud computing systems. The criteria for defining the taxonomy is based on the core ideas of distributed systems for massive data processing.

On the basis of proposed taxonomy and technical studies, the authors have evaluated the different cloud computing systems to provide necessary information that can help in future for the new development and improvement on existing systems. The proposed taxonomy provides researcher and developer the ideas on the current cloud systems, hype and challenges.

2.2 Open Source Cloud Management Platforms

X. Wen *et al.* [9] have described the function of OpenStack and OpenNebula briefly, and then compared them from provenance, architecture, hypervisors, security and other angles in detail. Moreover, the authors have provided some deployment recommendations according to different user demands and platform characteristics.

After the detailed analysis, the authors have concluded that the OpenStack can offer the same services as Amazon and be the alternative Amazon to users who don’t want to use a billing cloud. Thus, it is more suitable platform for enterprises as they can acquire the resources dynamically as well as encapsulate its services as some applications. Also, OpenNebula is well suitable for dealing with large amounts of data, leveraging existing

IT infrastructures and avoiding vendor lock-in. Thus, it is suitable for research institutes, universities and enterprises especially large data centers which want to build an open, flexible and scalable cloud to support their research. It is also ideal for users who want to run a number of cloud machines quickly.

P. Sempolinski *et al.* [8] have conducted a detailed study and comparative analysis of Eucalyptus, OpenNebula and Nimbus. The authors have described how these cloud management frameworks are related to other software components, required to create a functioning cloud computing system. They have also discussed some of the common challenges that emerge in setting up any of these frameworks and suggest avenues of further research and development.

The authors have identified strong tendencies in the each of the framework and finally concluded that Eucalyptus is geared toward a private company that wants its own cloud for its own use and wants to protect themselves from user malice and mistakes. OpenNebula is geared toward persons interested in the cloud or VM technology at their own end. It is also ideal for anyone who wants to set up just a few cloud machines quickly. Nimbus looks toward the more cooperative scientific community that might be less interested in the technical internals of the system, but has broad customization requirements.

A. Pillai [19] has analyzed the characteristics, architecture and applications of some of the most popular open source cloud computing platforms like Eucalyptus, OpenStack, Nimbus and OpenNebula. The main objective is to perform a comparative analysis of various cloud computing platforms to facilitate novice users to choose an open cloud platform depending on their requirements such as cloud types, interfaces, compatibility, implementation, deployment requirement, and development support.

The study also identifies challenges common to the platform and areas for improvement. For the further enhanced enhancement in open source platforms, it is suggested to incorporate more features to improve their framework.

C.Yang *et al.* [20] have proposed a scheme to build video services on cloud environment, which integrates KVM and OpenNebula open sources to provide a cloud virtual environment for end users. The proposed work integrates both Hadoop distribute file system (HDFS) and Nutch Search Engine in KVM-based cloud computing environment with video streaming related applications. This system can perform distributed computation in Map-Reduced programming in order to sufficiently shorten the time spent in searching indexes space construction. It uses IaaS and PaaS to construct a feature of cloud computing, which is a profound video cloud service with distributed file storage and a searching engine.

The system proposed by the authors is developed by applying open source; OpenNebula is used to build the IaaS environment, Hadoop is used as PaaS, and the application of cloud computing is realized by SaaS. As an IaaS provider, the proposed framework applies the idea of virtualization in the cloud system to economize power, web interfaces and user friendly ways to manage the virtual machines.

2.3 Security Challenges in Cloud Computing

H. Tianfield [10] has presented a comprehensive study on the challenges and issues of security in cloud computing. The author has firstly discussed the impacts of the distinctive characteristics of cloud computing, namely, multi-tenancy, elasticity and third party control, upon the security requirements. Further, he has analyzed the cloud security requirements in terms of the fundamental issues, i.e., confidentiality, integrity, availability, trust, and audit and compliance. Furthermore, the taxonomy for security issues in cloud computing is discussed.

Wadhawan *et al.* [21] have presented an elaborated study of security holes associated with Iaas implementation and their countermeasures. Infrastructure as a Service (IaaS) serves as the foundation layer for the other delivery models, and a lack of security in this layer also affects the other delivery models. The authors have proposed a Security Model for IaaS (SMI) as a guide for assessing and enhancing security in each layer of IaaS delivery model. The proposed model consists of three sides: IaaS components, security model, and the restriction level. This model indicates the relation between IaaS

components and security requirements, and eases security improvement in individual layers to achieve a total secure IaaS system.

A. Behl *et al.* [11] have investigated the problem of security from the cloud architecture perspective, the cloud characteristics perspective, cloud delivery model perspective and the cloud stakeholder perspective. The main objective is to identify the various attack vectors and security issues pertinent to cloud models and thus, present a solution to the various cloud security issues. They have proposed to adopt an adaptive approach in tackling the cloud security issues which will help in problem abstraction and capturing of security requirements of different stakeholders at different levels of details.
For the further enhancement in this research, data can be collected from various stakeholders such that various cloud models with their restrictions and merits can be adapted to security issue mitigation techniques which are neither stagnant nor on time.

D. Chen *et al.* [22] have provided a concise but all-round analysis on data security and privacy protection issues associated with cloud computing across all stages of data life cycle. The authors have discussed the various security issues linked with the data life cycle process from generation to destruction of the data. The main objective is to design a set of unified identity management and privacy protection frameworks across applications or cloud computing services.
On the basis of the analysis of security and privacy issues, the authors have determined that it is necessary to separate the sensitive data and access control, in order to achieve the cloud security. It is also suggested to have an integrated and comprehensive security solution to meet the needs of defense in depth. Moreover, authorization and access control mechanisms need to achieve a unified, reusable and scalable access control model and meet the fine grained access authorization.

2.4 Security Architecture for Cloud Storage

K. Nafi *et al.* [23] have proposed new security architecture for exchanging information in a cloud computing environment. In order to ensure a secure communication process, the proposed architecture includes AES file encryption system, RSA system for secure

communication, onetime password to authenticate users and MD5 hashing for hiding information. The main objective behind the proposed model is to solve main security issues like malicious intruders, hacking, etc. in cloud computing platform. Additionally, distributive server concept is used, thereby ensuring higher security for the whole cloud.
Experimental results have also been shown to determine the efficiency of the proposed model. From the comparative analysis, it is demonstrated that the proposed model works smoothly and ensures higher security than other present running models in a cloud computing environment.
Though the proposed architecture is highly secure, but RSA encryption system becomes fragile in long run process. Thus, there is a need to find out a light and secure encryption system for secure communication system and hiding information from others.

D. Mukhopadhyay *et al*. [24] proposed a methodology to overcome security threats which emerge with shared spaces in the cloud environment. The proposed framework involves securing of files through file encryption mechanism. The file present on the device is encrypted using password based AES algorithm. The user can download any of the uploaded encrypted files and read it on the system after the successful decryption. The suggested scheme aims at preventing every possible attack on the user data existing on the cloud. The integrity and confidentiality of the data uploaded by the user is ensured doubly by not only encrypting it but also providing access to the data only on successful authentication.
The authors have implemented the proposed framework to encrypt only text files which can be further enhanced to encrypt the audio and video files. Additionally, this framework can also be integrated with the social networking sites to exchange data securely in its encrypted form.

P. Rewagad *et al.* [25] have proposed a scheme for data security in clouds using a combination of authentication technique and key exchange algorithm blended with an encryption algorithm. The proposed framework makes use of digital signature and Diffie Hellman key exchange blended with (AES) Advanced Encryption Standard encryption algorithm to protect confidentiality of data stored in cloud. It is a three ways protection

scheme wherein digital signature provides authentication, encryption algorithm provides session encryption key and is used to encrypt data file as well, which is to be saved in cloud and lastly the trusted computing to verify the integrity of user data.
The strength of the proposed work is the unique combination which makes it tough for hackers to crack the security system, thereby protecting data stored in cloud.

U. Somani *et al.* [26] have assessed the cloud storage methodology and various security challenges associated with cloud computing. With the aim of the securing the data in the cloud, authors have proposed a concept of digital signature with RSA algorithm, to encrypt the data while transferring it over the network. This technique solves the dual problem of authentication and security. The strength of their work is the framework proposed to address security and privacy issue.

A. Mohta *et.al* [27] has presented a way to implement Third Party Auditor (TPA) who not only checks the reliability of Cloud Service Provider (CSP) but also checks the consistency and accountability of data. The author has also discussed the challenges of open issue of integrity and data dynamics. The main objective is to solve the problem of data privacy, accountability and integrity of data.
The proposed model involves the clients, cloud service provider and TPA. The suggested scheme involves retrieval of file, encryption and decryption of file, integrity checking from CSP and procedure for giving control to TPA. The responsibility of the TPA is to audit the cloud data storage efficiently. CSP provides service to client on successful authentication. Files are encrypted and decrypted using RSA algorithm. Message digest is generated using SHA-512 algorithm. After performing the desired file operation, clients send the data to CSP and TPA. The proposed scheme maintains the consistency at cloud data storage for CSP and client and also checks the integrity of data.

2.5 Security of Files through Encryption

Y.P. Singh *et al.* [28] have proposed a scheme that combines the security of a document by hybrid encryption method and authenticity by digital signatures. IDEA-RSA algorithm is used for hybrid cryptosystem and RSA digital signature algorithm is used to obtain

digital signature. The proposed joint signature scheme uses "encrypt-then-sign" instead of, "sign-then-encrypt ". The scheme has all the features of symmetric, asymmetric algorithm and digital signature.

First, a secret IDEA key of length 128 bits is generated. The message is encrypted in a quick manner using IDEA key. Afterwards, the key itself is encrypted with the recipient's public key using RSA encryption Algorithm. The public key is 1024 bit long which ensures the security. Thereafter, SHA-256 is used to obtain 256-bit condensed version of encrypted data which is further signed using RSA Digital signature algorithm to generate the digital signature. It is worth mentioning that time is not an issue for the proposed hybrid cryptographic system as the IDEA key consists of only 128 bits in comparison to 1024 bit of RSA. Thus, time taken to encrypt key is in milliseconds even with the slow RSA Cipher.

The system proposed by the authors has been designed and developed using C#. The effectiveness and correctness of the proposed scheme is illustrated through implementation and its results. Through the experimental results, it is determined that the proposed scheme has good throughput in contrast to RSA Algorithm, but relatively less than IDEA Algorithm.

The cryptographic system can be further enhanced by adding another layer of security for hiding the symmetric IDEA key. Instead of 128 bit IDEA key, cipher IDEA key can be passed to RSA system which can prevent the access to the valid key even if the RSA key is leaked.

A. Mathur [15] has proposed an algorithm for data encryption and decryption which is based on ASCII values of characters in the plaintext. The length of the encryption key must be equal to the length of plain text to be encrypted. The proposed algorithm is a kind of symmetric encryption algorithm because the decryption key can be generated from the encryption key by slightly modifying it.

Though the algorithm demonstrates an effective approach for data hiding but it fails to work for every combination of characters in 8-bit ASCII character set. The limitations of proposed algorithm can be overcome by further enhancing the approach so that it can encrypt and decrypt every possible combination of ASCII character set.

Chapter 3

Problem Statement

Previous chapter discussed related works and techniques available for data security on clouds. This chapter focuses on problem statement taken up in the thesis.

3.1 Gap Analysis

Cloud computing has become the next generation architecture of IT Enterprise. In contrast to the traditional architecture, cloud computing moves the application software and databases to the large data centers, where the management of the data and services may not be fully trustworthy. Although cloud architecture has its own security, but there exists an environment where the remote server can't be trusted and it is necessary to assure that the data has not been tampered with. Hence, it is extremely essential for the cloud to be secure.

In literature survey, various related works on security architecture for cloud storage are reviewed. On the basis of literature survey, following gaps are drawn:

i) The existing cloud security models are breached at some point of time and thus, are unable to provide an effective solution for security of data stored on the cloud servers.

ii) There is an emerging need of a security model that provides the maximum security by providing integrity, confidentiality and authentication.

3.2 Objectives of Thesis

The objectives of thesis are as follows:

i) Proposing a Two-tier hybrid cryptosystem coupled with digital signature scheme.

ii) Installing cloud environment using OpenNebula framework.

iii) Implementing proposed technique on the cloud environment.

From the perspective of data security, which is an important aspect of quality of service, cloud computing inevitably poses new challenging security threats. Integrity is the core process to ensure storage security in cloud. In case of compromise by service provider at

any cost, entrusting cloud side is of no use. Since the remote server can't be trusted completely, it is necessary to take measures at the user's end to ensure security and integrity. This can be accomplished by encrypting the data using an efficient cryptographic algorithm before uploading or transmitting it and decrypting it accordingly with the valid decryption key while accessing. Thus, even if an intruder gets access to the stored data, it does not make any sense as the relevant information cannot be accessed without corresponding key. Hence, if the data stored in the cloud is in encrypted form, it would effectively solve various security issues.

We have undertaken this problem of data security in our research to provide some solution correlated with file security in cloud computing storage environment.

Detailed design and algorithm for problem which has been formulated in chapter 3 are discussed in chapter 4 and chapter 5, respectively.

Chapter 4

OpenNebula

4.1 Overview of OpenNebula

OpenNebula is an open-source toolkit which aims to provide a flexible, open, scalable and comprehensive management layer to manage and simplify the operation of big data centers and transform existing infrastructure into an IaaS cloud [9]. It is a virtual infrastructure engine that enables the dynamic deployment and reallocation of virtual machines in a pool of physical resources. It extends the benefits of virtualization platforms from a single physical resource to a pool of resources, decoupling the server from both the physical infrastructure and the physical location. It offers the most feature-rich, customizable solution to build virtualized enterprise data centers and private cloud infrastructures on Xen, KVM and VMware deployments, and provides cloud consumers with choice of interfaces, from open cloud to de-facto standards, like the EC2 API. OpenNebula contains one front end and multiple back ends. According to allocation policies, it orchestrates storage, network, virtualization, monitoring, and security technologies to enable dynamic placement of multi-tier services (groups of interconnected virtual machines) on distributed infrastructures, combining both data center resources and remote cloud resources [20].

The main difference between OpenNebula and other commercial cloud solutions is that it guarantees complete interoperability with every existing infrastructure component already available. Thus, it avoids the vendor lock-in using common open industrial standards, such as EC2 API and Open Cloud Computing Interface (OCCI). Unlike other open source alternatives, OpenNebula does not embrace a particular hypervisor. It also does not have any specific infrastructure requirements, fitting well into any pre-existing environment, storage, network, or user-management policies. The plugin model, on which OpenNebula is implemented, give system integrators the ability to customize every aspect including virtualization, storage, information, authentication, authorization, and remote cloud services. [29].

4.2 Architecture of OpenNebula

OpenNebula has been designed to be modular in order to allow its integration with as many different hypervisors and environments as possible [30]. The OpenNebula internal architecture can be divided into three layers: Tools, Core and Drivers as depicted in the figure below [32]:

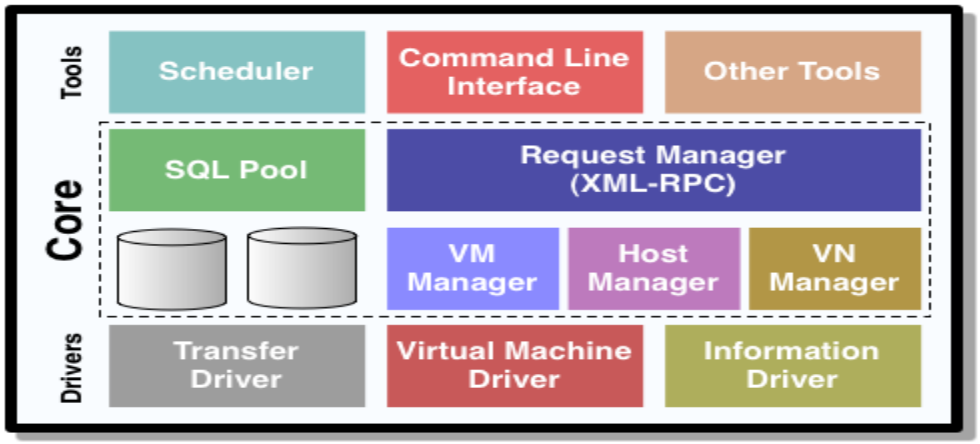

Figure 4.1: Architecture of OpenNebula [31]

4.2.1 Tools

This layer contains tools distributed with OpenNebula, such as the CLI, the scheduler, the libvirt API implementation and also 3rd party tools that can be easily created using the XML-RPC interface or the new OpenNebula Cloud API. They are explained as below:

i) *Command Line Interface:* A CLI for infrastructure administrators and users is provided with OpenNebula to manually manipulate the virtual infrastructure

ii) *Scheduler:* The Scheduler is an independent entity in the OpenNebula architecture, so it can be easily tailored or changed since it is decoupled from the rest of the components. It uses the XML-RPC interface provided by OpenNebula to invoke actions on virtual machines. The scheduler distributed with OpenNebula allows the definition of several resource and load aware policies.

4.2.2 OpenNebula Core

The core consists of a set of components to control and monitor virtual machines, virtual networks, storage and hosts. The core performs its actions (e.g. monitor a host, or cancel a VM) by invoking a suitable driver. The main functional components of OpenNebula core are explained as below:

i) *Request Manager:* The Request Manager exposes a XML-RPC Interface, and then depending on the invoked method a given component is called internally. The XML-RPC decouples most of the functionality in the OpenNebula core, from external components i.e. the Scheduler.

ii) *Virtual Machine Manager:* This Virtual Machine Manager (VMM) is responsible for the management and monitoring of VMs. The operations of the VM Manager are abstracted from the underlying hypervisor the use of pluggable drivers.

iii) *Transfer Manager:* The Transfer Manager (TM) is in charge of all the files transfers needed for the correct deployment of virtual machines. This includes the transfer of images to the cluster node selected for running the images of virtual machine, the transfer of the image from the cluster node to the image repository, the transfer of checkpoint files between cluster nodes for cold migrations or to the cluster front-end when the virtual machine is stopped, etc.

iv) *Virtual Network Manager:* The Virtual Network Manager (VNM) is responsible for the handling of IP and MAC addresses, allowing the creation of virtual networks by keeping track of leases (a set form by one IP and one MAC valid on a particular network) and their association with virtual machines and the physical bridges the VM are using.

v) *Host Manager:* The Host Manager (HM) manages and monitors the physical hosts. Monitor and management actions are performed through a suitable driver. The host monitoring infrastructure is flexible and can be extended to include any host attribute.

vi) *Database:* A persistent generic pool based on a SQLite3 backend is the core component of the OpenNebula internal data structures. This component provides OpenNebula with the scalability and reliability needed in the management of VMs.

4.2.3 Drivers

OpenNebula has a set of pluggable modules to interact with specific middleware (e.g. virtualization hypervisor, cloud services, file transfer mechanisms or information services), these adaptors are called drivers [29]. They are explained as follows:

i) *Transfer drivers:* These are used to manage the disk images on the current storage system—a shared one, such as Network File System (NFS) or on a non-shared one such as a simple copy over Secure Shell (SSH).
ii) *Virtual Machine drivers:* These are hypervisor-specific and are used for managing the virtual machine instances on the current hosts.
iii) *Information drivers:* These are used to retrieve the current status of virtual machine instances and hosts. They are hypervisor-specific, too—they are copied and remotely executed in every physical host through SSH.

4.3 Virtualization Providers for OpenNebula

The Virtualization subsystem is the component in charge of talking with the hypervisor installed in the hosts and taking the actions needing for each step in the VM lifecycle. The various hypervisors compatible with OpenNebula are as follows [1, 29]:

i) *Xen:* The first adopted OpenNebula hypervisor is Xen. It has been a unique leading open source virtualization technology for many years. Besides its use as a hypervisor in OpenNebula, Xen is also used standalone by many internet hosting companies such as Amazon EC2, Linode, and Rackspace Cloud. It was originally distributed as a Linux patchset, but is nowadays included in main GNU/Linux distributions such as SuSe, RedHat, and Debian.
ii) *KVM:* KVM (kernel-based virtual machine) is a complete virtualization technique for linux. It offers full virtualization, where each virtual machine interacts with its own virtualized hardware. In KVM design, KVM by itself is only an interface available to user space programs that can be called through the /dev/kvm special system file. For similar reasons, another open source project has been ported to support the KVM interface in gaining a full virtualization environment.

iii) *VMWare:* The VMware Infrastructure API (VI API) provides a complete set of language-neutral interfaces to the VMware virtual infrastructure management framework. By targeting the VI API, the Open- Nebula VMware drivers are able to manage various flavors of VMware hypervisors: ESXi (free), ESX and VMware Server.

4.4 Components of OpenNebula

OpenNebula assumes that your physical infrastructure adopts a classical cluster-like architecture with a front-end, and a set of cluster nodes where VMs will be executed. There is at least one physical network joining all the cluster nodes with the front-end.

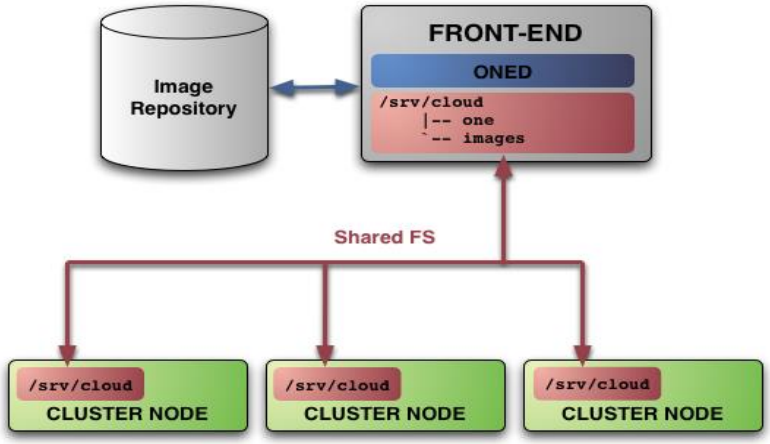

Figure 4.2: OpenNebula System [31]

The basic components of an OpenNebula system are [31]:

i) *Front-end:* The machine that holds the OpenNebula installation is called the front-end. This machine needs to have access to the storage medium and network connectivity to each host (node). It executes the OpenNebula and cluster services. Additionally, one should be able to ssh passwordlessly to all the hosts (including itself, the frontend) from the frontend.

ii) *Nodes:* The nodes are the hypervisor-enabled hosts which run VMs and provide the resources needed by the virtual machines. OpenNebula doesn't need to install any packages in the hosts, and the only requirements for them are ssh server

running and properly configured working hypervisor. Additionally one should be able to ssh passwordlessly to all the hosts (including itself and the frontend) from each host.

iii) *Image repository:* It is the storage medium that holds the base images of the VMs. VM images are registered, or created (empty volumes) in a image repository (datastore). When a VM is deployed the images are transferred from the datastore to the hosts. Each datastore has to be accessible through the front-end using any suitable technology NAS, SAN or direct attached storage.

iv) *OpenNebula daemon:* OpenNebula daemon (oned) is the core service of the system. It manages the life-cycle of the VMs and orchestrates the cluster subsystems (network, storage and hypervisors).

v) *Drivers:* These are the programs used by the core to interface with a specific cluster subsystem, e.g. a given hypervisor or storage file system.

4.5 Key Features of OpenNebula

The key features of OpenNebula are explained as below [31]:

i) *Powerful User Security Management:* OpenNebula offers a pluggable Auth subsystem for authentication and authorization of requests with complete functionality for user management. Any interface to OpenNebula communicates with the core using xml-rpc calls that contain the user's session string, which is authenticated by the OpenNebula core comparing the username and password with the registered users. Authorization framework allows multiple-role support for different types of users and administrators, delegated control to authorized users, secure isolated multi-tenant environments, and easy resource sharing.

ii) *Multi-tenancy with Group Management:* Administrators can group users into organizations that can represent different projects, divisions etc. Each group has configurable access to shared resources, thus, enabling a multi-tenant environment with multiple groups sharing the same infrastructure.

iii) *High Availability:* OpenNebula provides a persistent database backend with support for high availability configurations. Additionally by providing configurable behavior in the event of host, VM, or OpenNebula instance failure, it

provides an easy to use and cost-effective failover solution. It also extends support for high availability architectures.

iv) *Distributed Resource Optimization:* OpenNebula comes with a powerful and flexible requirement/rank matchmaker scheduler that implements rank schedule policy for providing automatic initial VM placement for the definition of workload and resource-aware allocation policies such as packing, striping, load-aware and affinity-aware. The goal of this policy is to prioritize those resources more suitable for the VM. There also exists resource quota management to allocate, track and limit resource utilization.

v) *Extension and Integration:* OpenNebula has a modular and extensible architecture which enables it to fit into any existing datacenter. It also offers customizable drivers for the main subsystems to easily leverage existing IT infrastructure and system management products such as virtualization, storage, network etc. Configuration and tuning parameters are available to adjust behavior of the cloud management instance to match the requirements of the environment and use cases.

4.6 Comparative Analysis with other Open Source Cloud Solutions

OpenNebula Cloud computing platform is better than other open source cloud solutions in the following ways [31]:

i) *Data Center Virtualization:* OpenNebula is more focused on data center virtualization and enterprise private cloud computing than on public AWS-like cloud features. It offers a flexible management platform that adapts to underlying processes and models for computing, storage, security, monitoring, and networking in their data centers. Other open-source solutions mainly focus on public cloud features, adopting a rigid management model dictated by cloud API semantic, and do not realize the full potential of virtualization in the data center.

ii) *Richer functionality and wider Integration capabilities:* OpenNebula does not only bring an open-source implementation of the most common public cloud interfaces, but also a much richer and flexible management model and the latest

innovations in data center virtualization for the deployment of enterprise clouds in the existing infrastructure. Other open-source cloud solutions are not fully vendor agnostic, providing a tighter integration and superior functionality on its company's hypervisor.

iii) *Production-proven and Packaged product:* OpenNebula comprises of all key functionalities for cloud computing, storage and networking with a single install. This reduces cost and complexity of the cloud and ensures its long term stability and performance through a single integrated patching and updating process, and one-stop support. Other open-source alternatives provide a set of technologies with different maturity levels that have to be integrated to build something functional.

iv) *Leverages power of User-driven development:* OpenNebula's roadmap is completely driven by user's needs with features that meet real demands, and not features that result from an agreement between IT vendors planning to create their own proprietary cloud solution. Many of our users are also active contributors to the software. OpenNebula has a fast development and release cycle to quickly react to changes and incorporate the most-demanded functionalities.

v) *Cloud API agnostic:* OpenNebula is not an open-source implementation of a popular cloud interface, but a powerful cloud management tool that provides cloud consumers with choice of interfaces, from open cloud to de-facto standards. OpenNebula has been always committed to the implementation of the main specifications from standards bodies.

4.7 Planning and Installation of OpenNebula

The various steps involved in the planning and installation of OpenNebula toolkit are discussed as below:

4.7.1 System Requirements

i) OneHost: [Server/Frontend]

The front end should satisfy the following system specifications:

a) VT-x enabled server hardware [Hostname : OneHost]

b) Atleast 100 GB HDD free space
c) Atleast 2 GB RAM.

ii) VMHost: [Cluster node]

The cluster node should satisfy the following system specifications:

a) VT-x enabled server hardware [Hostname : VMHost]
b) Atleast 100 GB HDD free space,
c) Atleast 2 GB RAM

{Note: One can also have more than one VM Host}

4.7.2 Preparing OneHost

The various steps involved in the preparation of OneHost are discussed as below:

i) Install Ubuntu Server 12.10 64 bit software in OneHost
ii) Have GRUB installed.
iii) Install OpenSSH server alone.
iv) Once, the installation is complete, edit /etc/sysctl.conf and uncomment the below line.
net.ipv4.ip_forward = 1
v) Edit /etc/rc.local and add the following lines just above the “exit 0” line
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
iptables -t nat -I POSTROUTING -s 172.31.4.149/24 -d 172.31.4.150/8 -j MASQUERADE
vi) Restart the Onehost machine.

4.7.3 Preparing VMHost

The various steps involved in the preparation of VMHost are discussed as below:

i) Install Ubuntu Desktop 12.04 32 bit.
ii) Install OpenSSH client.
iii) Once installation is over, login to VMHost using server console/SSH. One should be able to ping OneHost and connect to internet.

iv) Install bridge-utils using below command.

sudo apt-get install bridge-utils

v) Edit /etc/network/interfaces to add the below lines to add bridge configuration.

auto br0

iface br0 inet dhcp

bridge_ports eth0

bridge_fd 9

bridge_hello 2

bridge_maxage 12

bridge_stp off

vi) Restart the networking using the following command.

sudo /etc/init.d/networking restart

4.7.4 Configuring OneHost

The various steps involved in configuring OneHost are discussed as below:

i) Create a folder “cloud” and create a group named “cloud”.

sudo mkdir -p /srv/cloud/

sudo groupadd -g 10000 cloud

ii) Create a user “oneadmin”, add user to group “cloud” and have /srv/cloud/one as home folder.

sudo useradd -u 10000 -m oneadmin -d /srv/cloud/one -s /bin/bash -g cloud.

iii) Set up the password for “oneadmin” and make oneadmin owner of “/srv/cloud”.

sudo passwd oneadmin

sudo chown -R oneadmin:cloud /srv/cloud/

iv) Test by logging as user “oneadmin” and exit.

su -l oneadmin

exit

v) Install Network File Server [NFS]

sudo apt-get install nfs-kernel-server

vi) Edit /etc/exports and add the following line to make folder /srv/cloud/one shareable with VMHost

/srv/cloud/one 172.31.4.149/24 (rw, fsid=0, nohide, sync, root_squash, no_subtree_check)

vii) Restart NFS server

sudo /etc/init.d/nfs-kernel-server start

viii) Add VMHost's Hostname to /etc/hosts of OneHost

172.31.4.149 VMHost

ix) Create a SSH key for oneadmin and disable host key checking else make all host keys known on the OpenNebula node.

su -l oneadmin

ssh-keygen

ssh-copy-id -i ~/.ssh/id_rsa.pub <remote-host>

[add below two lines to SSH config file]

Host *

StrictHostKeyChecking no

{Note: ssh-copy-id appends the keys to the remote-host's .ssh/authorized_key}

x) Exit from editor and try pinging VMHost, it should ping.

4.7.5 Configuring VMHost

Before configuring VMHost check if you are able to ping OneHost IP (172.31.4.150) and OneHost's Gateway (172.31.4.1) from VM Host.

i) Add oneadmin to sudo group

usermod -a -G sudo oneadmin

ii) Install "NFS common" to enable access to the folder "/srv/cloud/one" of OneHost.

sudo apt-get update

sudo apt-get install nfs-common

iii) Edit /etc/fstab and add an NFS entry for /srv/cloud/one

172.31.4.150:/srv/cloud/one /srv/cloud/one nfs defaults 0 0

iv) Create folder structure /srv/cloud/one in VMHost and mount it as per "fstab" entry.

```
sudo mkdir -p /srv/cloud/one
sudo mount /srv/cloud/one
```

v) Create user "oneadmin" and group "cloud".

```
sudo groupadd -g 10000 cloud
sudo useradd -u 10000 -g cloud -m oneadmin -s /bin/bash
sudo usermod -d /srv/cloud/one oneadmin
sudo usermod -a -G cloud,root oneadmin
sudo passwd oneadmin
sudo chown oneadmin:cloud /srv/cloud
sudo mount /srv/cloud/one
mount
{Note: You should see the below line}
172.31.4.150:/srv/cloud/one on /srv/cloud/one type nfs (rw,addr=172.31.4.150)
```

vi) Install KVM hypervisor using the following command.

```
sudo apt-get install qemu-kvm libvirt-bin ubuntu-vm-builder bridge-utils ruby
```

vii) Libvirt needs to be configured to enable users of group "cloud" to manage the VM's and to allow VNC connections. Edit "/etc/libvirt/libvirtd.conf" and make the following changes:

```
unix_sock_group = "cloud"
```

viii) Edit /etc/libvirt/qemu.conf and uncomment vnc_listen line and restart libvirt

```
vnc_listen = "0.0.0.0"
sudo service libvirt-bin restart
```

ix) Configure libvirt to allow access from the members of group "cloud"

```
sudo chown  :cloud /var/run/libvirt/libvirt-sock
```

4.7.6 Installing and Configuring OpenNebula in OneHost

The various steps involved in installing and configuring OpenNebula in OneHost are discussed as follows:

i) Before installing OpenNebula, install all pre-requisite packages

sudo apt-get install libsqlite3-dev libxmlrpc-c3-dev g++ ruby libopenssl-ruby libssl- dev ruby-dev

sudo apt-get install libxml2-dev libmysqlclient-dev libmysql++-dev libsqlite3-ruby libexpat1-dev

sudo apt-get install rake rubygems libxml-parser-ruby1.8 libxslt1-dev genisoimage scons

sudo gem install nokogiri rake xmlparser

sudo apt-get install mysql-server

ii) Configure MySql support

mysql -uroot –preema {Note: Here 'root' is the user and 'reema' is the password}

Create user 'oneadmin'@'localhost' IDENTIFIED BY 'oneadmin';

Create Database opennebula;

Grant all priviliges on opennebula.* to 'oneadmin' identified by 'oneadmin';

quit;

iii) Before installing OpenNebula, configure mysql support.

cd ~/opennebula-3.1.90 [change your folder to opennebula source]

scons sqlite=no mysql=yes

iv) Install openebula in /srv/cloud/one accessible by group cloud and as user "oneadmin".

./install.sh -u oneadmin -g cloud -d /srv/cloud/one

v) Create a profile file (~/.bash_profile) to set Environment Variables required to start and use services rendered by "one".

nano ~/.bash_profile

export ONE_LOCATION=/srv/cloud/one

export ONE_AUTH=$ONE_LOCATION/.one/one_auth

export ONE_XMLRPC=http://localhost:2633/RPC2

export PATH = $ONE_LOCATION/bin:/usr/local/bin: /var/lib/gems/1.8/bin/: /var /lib /gems/1.8/: $PATH

vi) Execute the profile file and set the environment variables.

source ~/.bash_profile

vii) Create and store OpenNebula user (oneadmin) and password in a file.

mkdir ~/.one

echo “oneadmin:<Password>” > ~/.one/one_auth

viii) Make following changes in OpenNebula configuration file (~/etc/oned.conf).

nano ~/etc/oned.conf

a) Comment following line

#DB = [backend = “sqlite”]

b) Set SQL as mysql-uncomment #lines 61 through 66 or near by

DB = [backend = “mysql”,

server = “localhost”,

port = 0,

user = “oneadmin”,

passwd = “oneadmin”,

db_name = “opennebula”]

c) Add below lines just below first TM_MAD definition.

TM_MAD = [

name = “tm_nfs”,

executable = “one_tm”,

arguments = “tm_shared/tm_shared.conf”]

ix) Start Nebula

one start { Note: it should start with no error messages}

4.8 Administering OpenNebula

i) Adding a host

Checkpoints:

a) Check /etc/hosts file of OneHost and VMHost for hostname entries.

b) Try ssh VMHost from $ prompt and you should be able to login with no password.

Command: *onehost create <hostname> --im <im_mad> --vm <vmm_mad> --net dummy*

where,

im_mad specifies “Information drivers” - used to monitor the host.

vmm_mad specifies “Virtualization drivers” - used to manage VMs.

ii) Listing registered host(s)

Command: *onehost list*

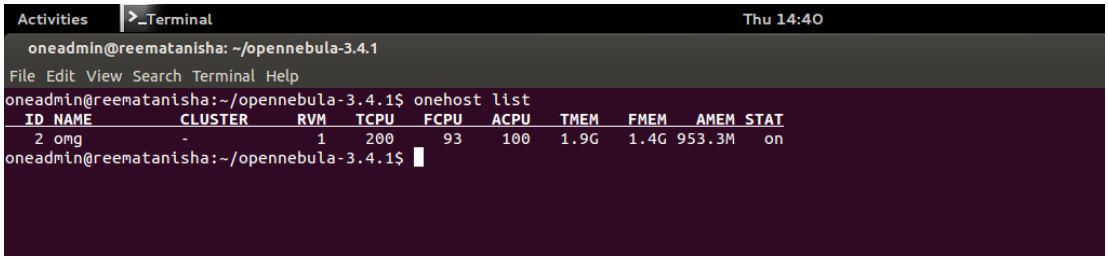

Figure 4.3: Registered Hosts

iii) Detailed information about the registered host

Command: *onehost show <host ID> /<host_Name>*

e.g onehost show 2

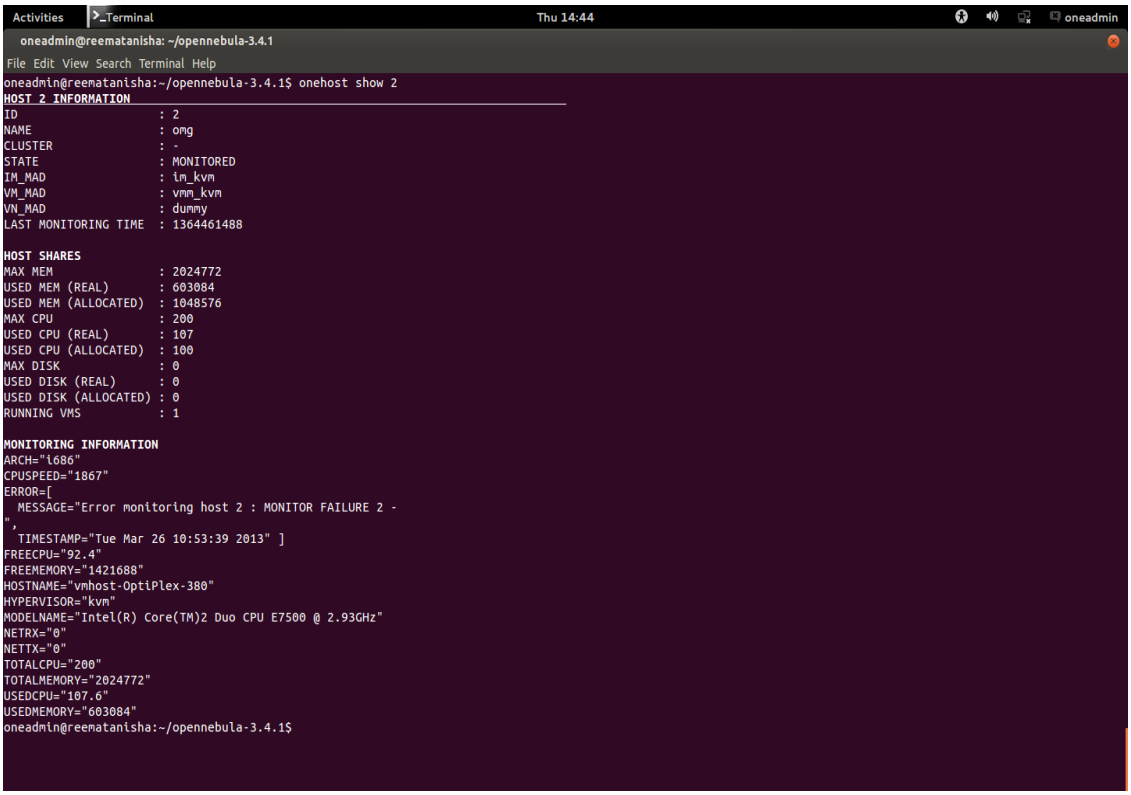

Figure 4.4: Detailed Information of Host

4.9 Creating a Virtual Machine

i) Configuring a filesystem datastore.

The first step to create a filesystem datastore is to set up a template file (ds.conf) for it. The datastore type is set by its drivers, in this case be sure to add DS_MAD=fs. The other important attribute to configure the datastore is the transfer drivers. These drivers determine how the images are accessed in the hosts. Various attributes of configuration file are explained as below:

Table 4.1: Attributes of Datastore Configuration File

Attribute	Description
NAME	The name of the datastore
DS_MAD	The DS type, use fs for the Filesystem datastore
TM_MAD	Transfer drivers for the datastore: shared, ssh or qcow2
SAFE_DIRS	If you need to un-block a directory under one of the RESTRICTED_DIRS. A space separated list of paths.

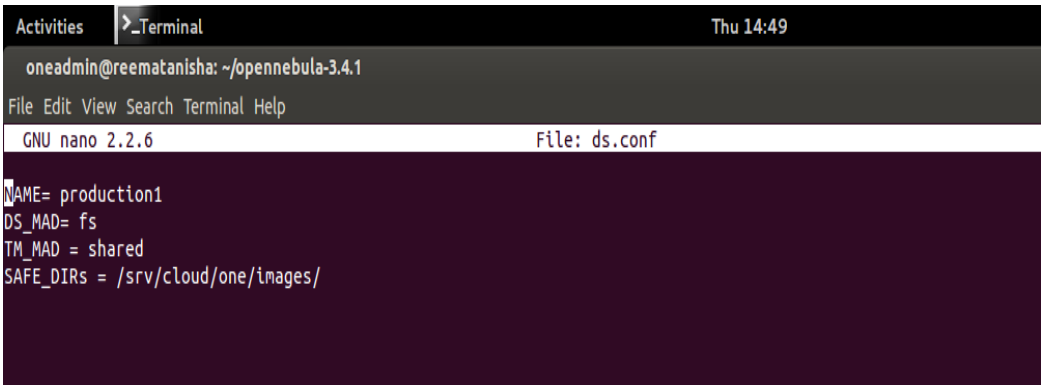

Figure 4.5: Datastore Configuration File

ii) Creating a datastore

Command: *onedatastore create ds.conf*

where,

ds.conf – Datastore configuration file

{Note: On successful execution of command, datastore will be created and an ID will be assigned}

iii) Listing available Datastores

Command: *onedatastore list*

Figure 4.6: List of Available Datastores

iv) Detailed information about the registered datastores

Command: *onedatastore show <Datastore ID>*

e.g onedatastore show 101

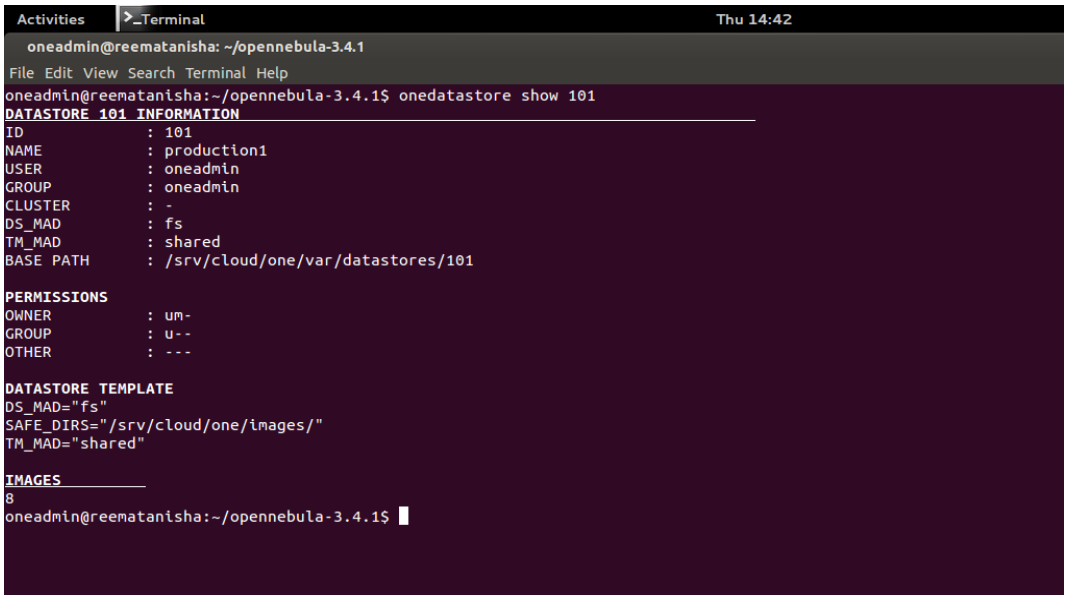

Figure 4.7: Detailed Information of Datastore

v) Configuring an image definition file

An OpenNebula image is a resource containing an operative system or data, to be used as a virtual machine disk. The first step is to define an image template file (vmImage.img). Various attributes of the template are explained as below:

Table 4.2: Attributes of Image Template

Attribute	Description
NAME	Name that the Image will get. Every image must have a unique name.
TYPE	Type of the image i.e. OS,CDBLOCK,DATABLOCK
PATH	Path to the original file that will be copied to the image repository.
PUBLIC	Public scope of the image. If omitted, the default value is NO.

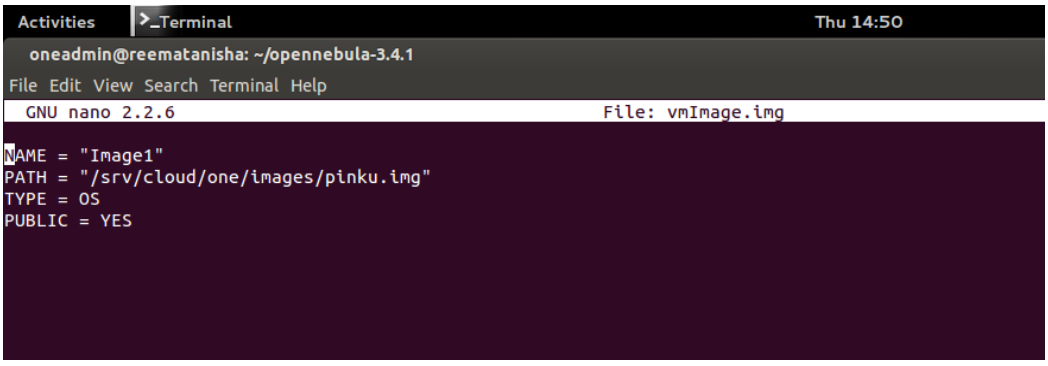

Figure 4.8: Image Definition File

vi) Submitting an image

After creating the image template, submit it using the *oneimage create* command.

e.g. oneimage create vmImage.img -d 101

where,

vmImage.img – Image template file

101 – Datastore ID

{Note: one successful execution of command, image will be created and an image ID will be generated}

vii) Listing available images

To check the available images in the repository, *oneimage list* command is issued.

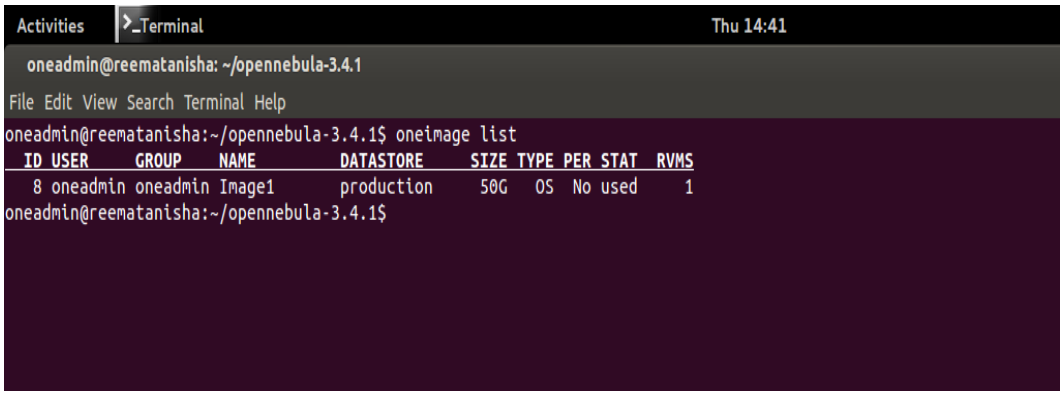

Figure 4.9: Images Available in Repository

viii) Detailed information about the registered images

Command: *oneimage show <Image ID>*

e.g oneimage show 8

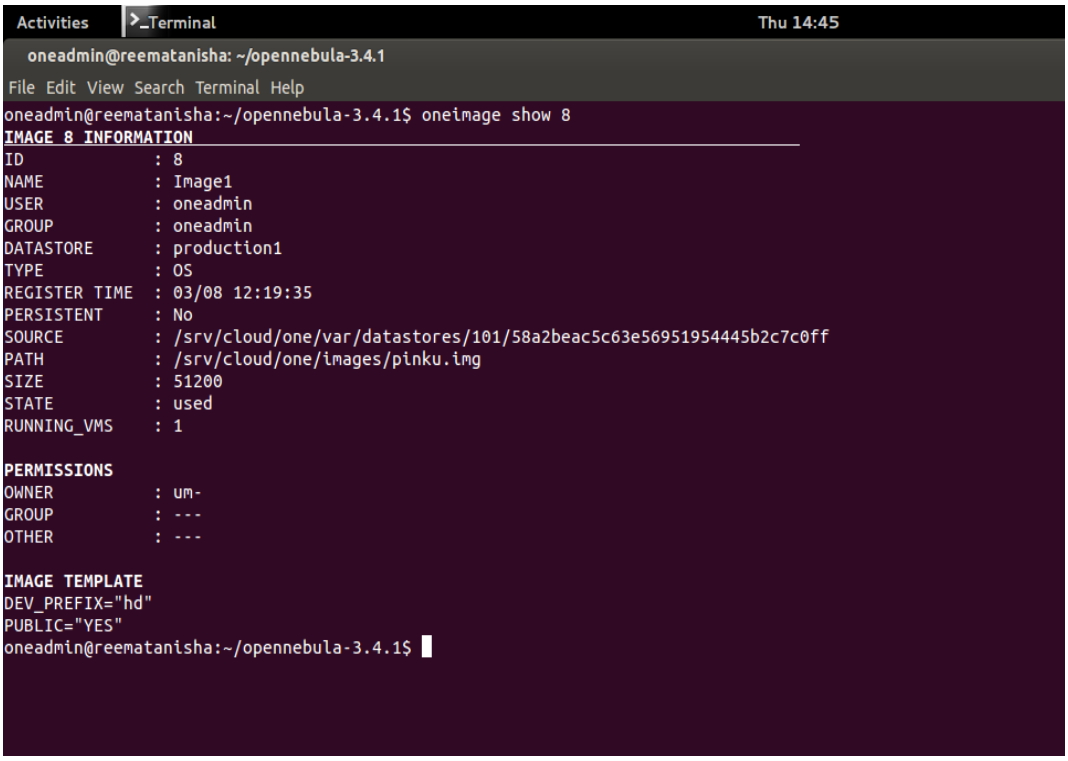

Figure 4.10: Detailed Information of an Image

ix) Configuring a Virtual Network (VN)

OpenNebula allows the creation of virtual networks by mapping them on top of the physical ones. All Virtual Networks are going to share a default value for the MAC prefix, set in the oned.conf file.

There are two types of Virtual Networks in OpenNebula:

i) Fixed, defines a fixed set of IP-MAC pair addresses

ii) Ranged, defines a class network.

Various attributes of network template are explained as below:

Table 4.3: Attributes of Network Template File

Attribute	Description
NAME	Name of the Virtual Network.
TYPE	Type of the virtual network
PUBLIC	Public scope of the image. If omitted, the default value is NO
BRIDGE	Name of the physical bridge
NETWORK_ADDRESS	Base network address to generate IP addresses.

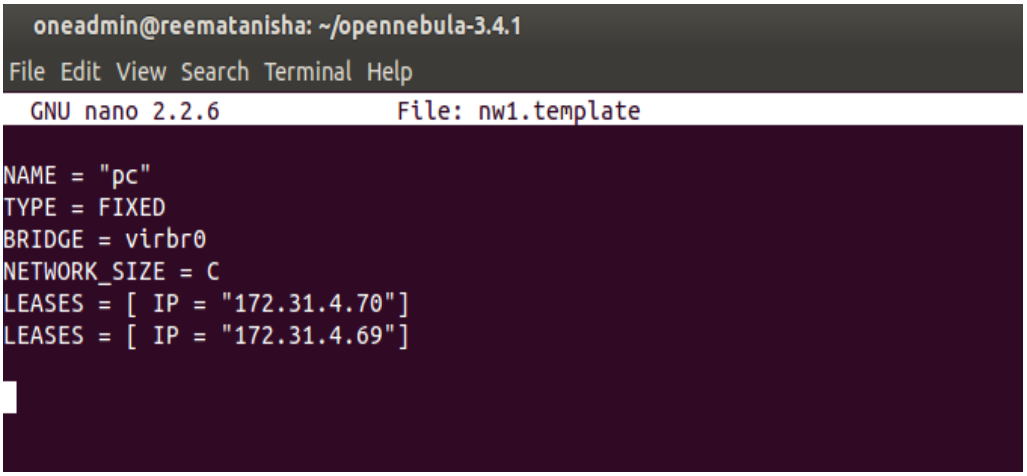

Figure 4.11: Network Template

x) Adding VN

Once a template for a VN has been defined, the *onevnet* command can be used to create a virtual network.

onevnet create nw1.template

where,

nw1.template - network template file

xi) Listing available VNs

Command: *onevnet list*

```
oneadmin@reematanisha: ~
File  Edit  View  Search  Terminal  Help
oneadmin@reematanisha:~$ onevnet list
  ID USER     GROUP    NAME              CLUSTER    TYPE BRIDGE   LEASES
   0 oneadmin oneadmin private cloud     -             F virbr0         0
   1 oneadmin oneadmin private cloud1    -             R virbr0         0
   3 oneadmin oneadmin Small network     -             F virbr0         0
   6 oneadmin oneadmin PRIVATE CLOUD2    -             F virbr0         1
   7 oneadmin oneadmin pc                -             F virbr0         2
   8 oneadmin oneadmin pc2               -             F virbr0         2
oneadmin@reematanisha:~$
```

Figure 4.12: List of Virtual Networks

xii) Detailed information about a specific VN

Command: *onevnet show <Network ID>*

e.g. onevnet show 7

```
oneadmin@reematanisha:~$ onevnet show 7
VIRTUAL NETWORK 7 INFORMATION
ID             : 7
NAME           : pc
USER           : oneadmin
GROUP          : oneadmin
CLUSTER        : -
TYPE           : FIXED
BRIDGE         : virbr0
VLAN           : No
PHYSICAL DEVICE:
VLAN ID        :
USED LEASES    : 2

PERMISSIONS
OWNER          : um-
GROUP          : ---
OTHER          : ---

VIRTUAL NETWORK TEMPLATE
NETWORK_SIZE="C"

USED LEASES
LEASE=[ IP="172.31.4.69", MAC="02:00:ac:1f:04:45", USED="1", VID="22" ]
LEASE=[ IP="172.31.4.70", MAC="02:00:ac:1f:04:46", USED="1", VID="26" ]
oneadmin@reematanisha:~$ clear
```

Figure 4.13: Detailed Information of a Network

xiii) Configuring a VM template file

The foremost step prior to deploying a VM is to define the template file (pc.one). A template file consists of a set of attributes that defines a VM.

Various attributes of template file are explained as below:

Table 4.4: Attributes of VM Template

Attribute	Definition
NAME	Name that the VM will get for description purposes
CPU	Percentage of CPU divided by 100 required for the Virtual Machine
NIC	Defines the network interface of a VM
GRAPHICS	Whether the VM should export its graphical display and how
IMAGE_ID	ID of the Image to use
MEMORY	Amount of RAM required for the VM, in Megabytes.
LISTEN	It is the sub-attribute of GRAPHICS. It determines which IP to listen on.

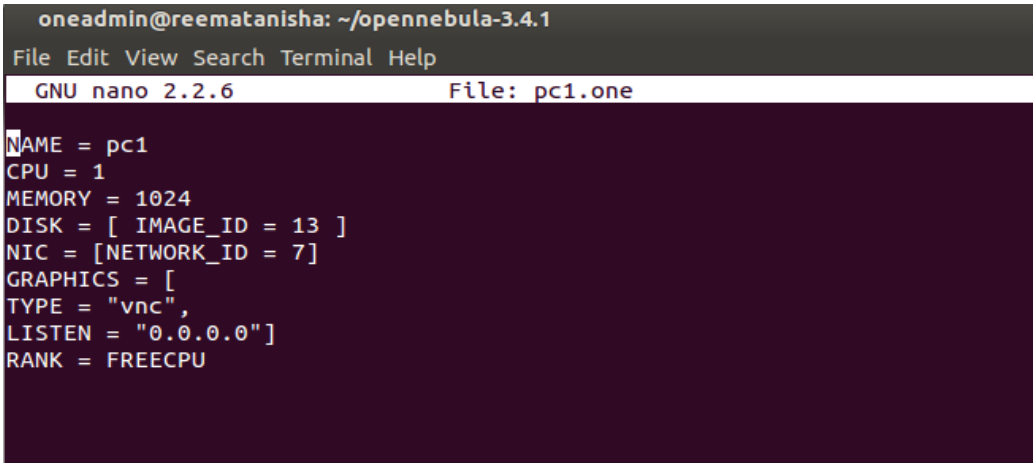

Figure 4.14: VM Template

xiv) Deploying a VM

Command: *onevm create <VM template file>*

e.g. onevm create pc1.one

where,

pc1.one – VM template file

xv) Listing deployed VM(s)

Command: *onevm list*

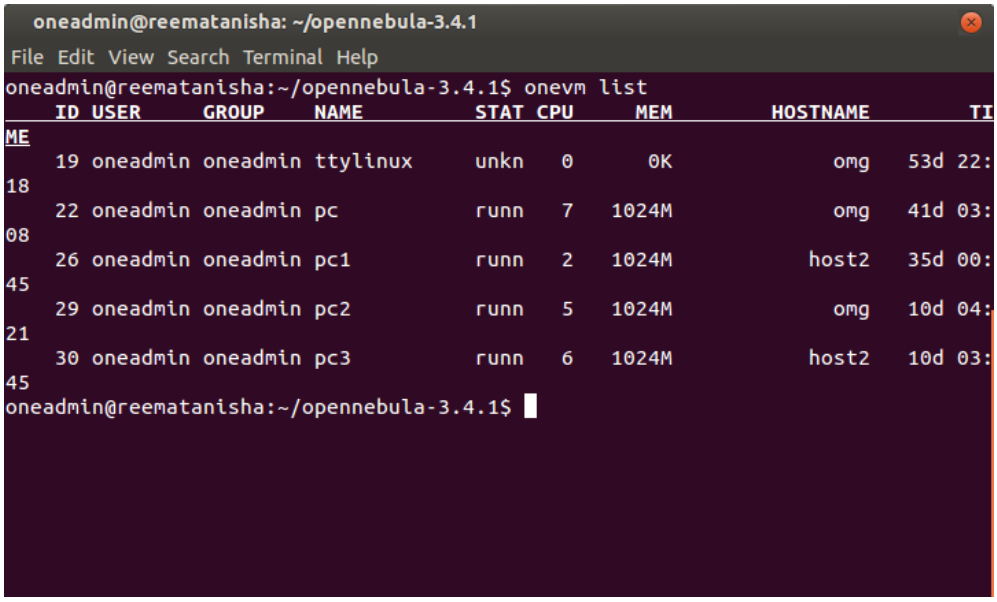

Figure 4.15: Listing Available Virtual Machines

xvi) Detailed information about VM

Command: *onevm show <VM ID>*

e.g. onevm show 26

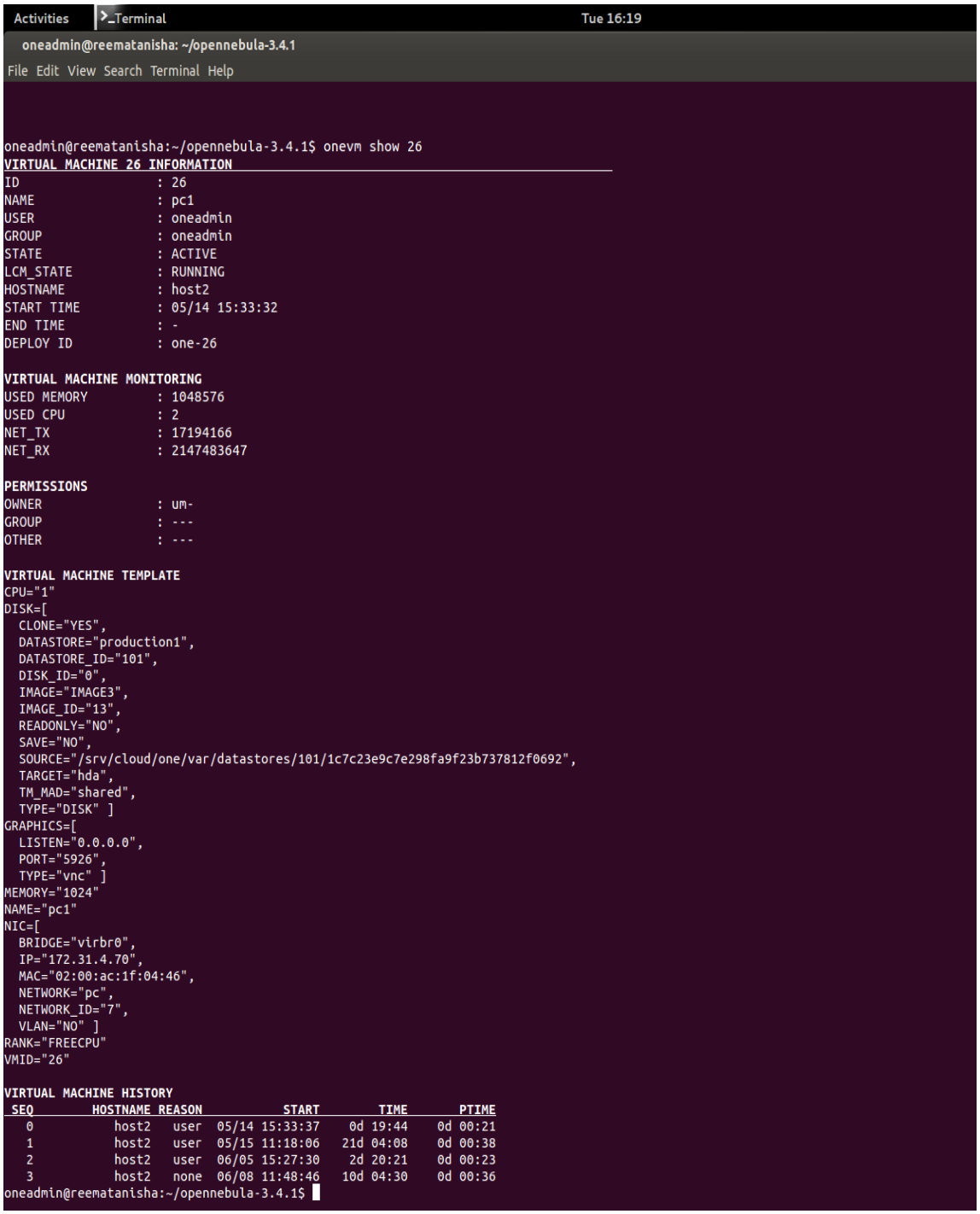

Figure 4.16: Detailed Information of VM

Chapter 5

Proposed Work

5.1 Key Scenario

Clients and data owners, when outsourcing data, are assumed to trust the remote cloud server to faithfully maintain outsourced data. The server is then relied upon for the availability of outsourced data, so the data owner and clients can access data whenever requested. However, server is not trusted with the confidentiality of the actual database content, as outsourced data may contain sensitive information. It is necessary to prevent the server from making unauthorized accesses to the database. For this purpose, the data owner encrypts her data and sends the encrypted data to the server for storage. For the purpose of hiding the sensitive information before outsourcing the data files to the remote cloud server, a two-tier hybrid cryptosystem coupled with the digital signature scheme is proposed.

5.2 Encryption Algorithms Used

The proposed hybrid cryptosystem involves the following algorithms:

i) IDEA Algorithm
ii) Improved ASCII Code Based Encryption Algorithm
iii) RSA Algorithm
iv) RSA Digital Signature Scheme

5.2.1 IDEA Algorithm

The IDEA is a block cipher that operates with 64-bit plaintext and cipher text blocks and is controlled by a 128-bit key. The algebraic idea behind IDEA is mixing of three incompatible algebraic operations on 16-bit blocks: bitwise XOR, addition modulo 2^{16}, and multiplication modulo $2^{16} + 1$ [32]. The algorithm structure has been chosen such that, with the exception that different key sub-blocks are used, the encryption process is identical to the decryption process. The algorithm consists of eight identical rounds and a "half round" final transformation.

5.2.1.1 Key Scheduling

Each of the eight complete rounds requires six sub-keys, and the final transformation "half round" requires four sub-keys; so, for the entire process 52 (8 x 6 + 4) different 16-bit sub-blocks have to be generated from the 128-bit key.

Encryption Keys

The 52 sub-blocks of 16-bit key which are generated from the 128-bit key are produced as follows [33]:

i) First, the 128-bit key is partitioned into eight 16-bit sub-blocks which are then directly used as the first eight key sub-blocks.
ii) The 128-bit key is then cyclically shifted to the left by 25 positions, after which the resulting 128-bit block is again partitioned into eight 16-bit sub-blocks to be directly used as the next eight key sub-blocks.
iii) The cyclic shift procedure described above is repeated until all of the required 52 16-bit key sub-blocks have been generated.

Decryption Keys

Decryption Keys are generated from the already generated encryption keys.

The decryption keys for the first round are generated as described below [33]:

$K_{(i,j)}$ denotes the j^{th} decryption key of decryption round i.

$Z_{(i,j)}$ denotes the j^{th} encryption key of encryption round i.

$K_{(1,1)} = (Z_{(9,1)})^{-1}$, where $(Z_{(9,1)})^{-1}$ denotes the inverse multiplication modulo $2^{16} + 1$ of the first encryption key of encryption round 9 i.e. the "half round" final transformation ; $K_{(1,2)} = -Z_{(9,2)}$, where $-Z_{(9,2)}$ denotes the inverse addition modulo 2^{16} of the second encryption key of encryption round 9 ; $K_{(1,3)} = -Z_{(9,3)}$; $K_{(1,4)} = (Z_{(8,4)})^{-1}$; $K_{(1,5)} = Z_{(8,5)}$; and $K_{(1,6)} = Z_{(8,6)}$.

The remaining decryption keys are generated similarly for the complete decryption rounds. The decryption keys for the final transformation "half round "are: $K_{(9,1)} = (Z_{(1,1)})^{-1}$, $K_{(5,2)} = -Z_{(1,2)}$, $K_{(9,3)} = -Z_{(1,3)}$, and $K_{(9,4)} = (Z_{(1,4)})^{-1}$.

5.2.1.2 Encryption Process

The 64 bit plain text is divided into four 16 bit sub- blocks. Each round uses each of the three algebraic operations: bitwise XOR, addition modulo 2^{16}, and multiplication modulo $2^{16} + 1$.

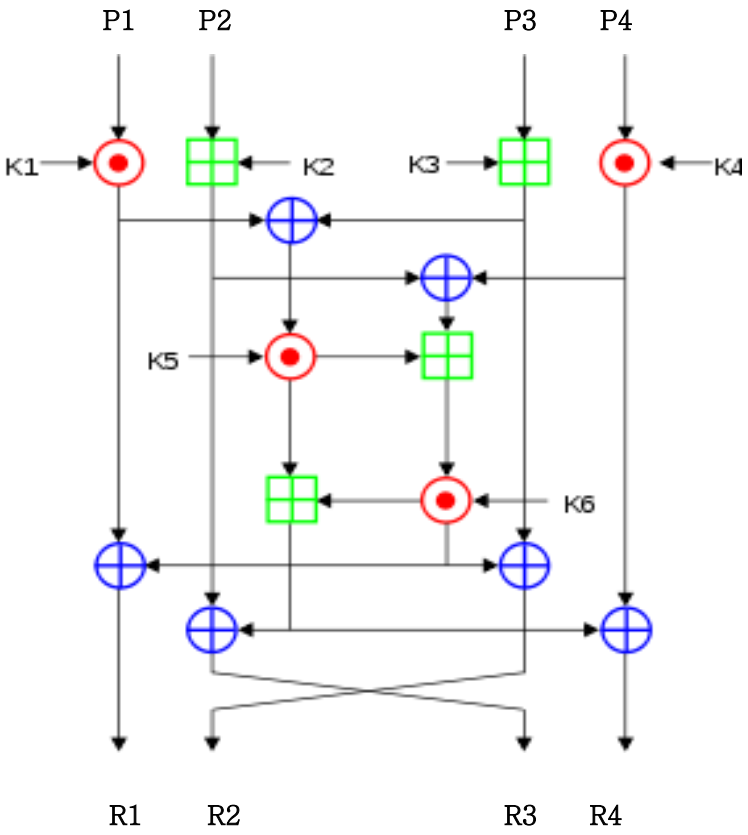

Figure 5.1: One Round of IDEA [34]

Here are the fourteen steps of a complete round.

i) Multiply P1 and the first sub key K1.

ii) Add P2 and the second sub key K2.

iii) Add P3 and the third sub key K3.

iv) Multiply P4 and the fourth sub key K4.

v) Bitwise XOR the results of steps 1 and 3.

vi) Bitwise XOR the results of steps 2 and 4.

vii) Multiply the result of step 5 and the fifth sub key K5.

viii) Add the results of steps 6 and 7.

ix) Multiply the result of step 8 and the sixth sub key K6.

x) Add the results of steps 7 and 9.

xi) Bitwise XOR the results of steps 1 and 9.

xii) Bitwise XOR the results of steps 3 and 9.

xiii) Bitwise XOR the results of steps 2 and 10.

xiv) Bitwise XOR the results of steps 4 and 10.

At the end of the first encryption round four 16-bit values are produced which are used as input to the second encryption round. The process described above is repeated in each of the subsequent 7 encryption rounds using different 16-bit key sub-blocks for each combination. During the subsequent output transformation, the four 16-bit values produced at the end of the 8^{th} encryption round are combined with the last four of the 52 key sub-blocks using addition modulo 2^{16} and multiplication modulo $2^{16} + 1$ to form the resulting four 16-bit cipher text blocks.

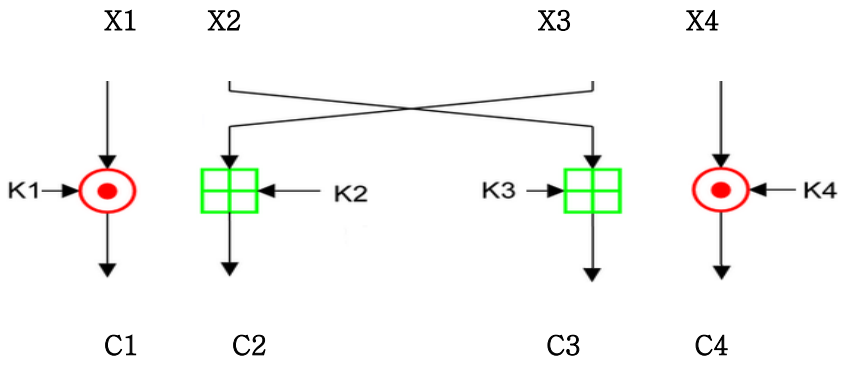

Figure 5.2: Output Round [34]

Here are the four steps for output round.

i) Multiply X1 and the first sub key.

ii) Add X2 and the second sub key.

iii) Add X3 and the third sub key.

iv) Multiply X4 and the fourth sub key.

5.2.1.3 Decryption Process

The computational process used for decryption of the cipher text is essentially the same as that used for encryption of the plaintext. The only difference compared with encryption is that during decryption, different 16-bit key sub-blocks are generated.

More precisely, each of the 52 sub-blocks of 16-bit key used for decryption is the inverse of the key sub-blocks used during encryption in respect of the applied algebraic group operation. Additionally, the key sub-blocks must be used in the reverse order during decryption in order to reverse the encryption process.

5.2.2 RSA Algorithm

The Rivest-Shamir-Adleman (RSA) cryptosystem is one of the best known public key cryptosystems for key exchange or digital signatures or encryption of blocks of data. RSA uses a variable size encryption block and a variable size key [35]. The RSA cryptosystem encrypts and decrypts a message using a pair of keys known as public key and private key.

5.2.2.1 Key Generation

In the steps described below, 'n' denotes the modulus, 'e' denotes the encryption exponent , 'd' denotes the secret exponent or decryption exponent.

i) Choose two distinct large random prime numbers p and q.

ii) Compute n = p * q, where n is used as the modulus for both the public and private keys.

iii) Compute phi (n) = (p -1) * (q-1)

iv) Choose an integer e such that 1 < e < phi(n) , and e and phi(n) share no factors other than 1, where e is released as the public key exponent.

v) Compute d to satisfy the congruence relation d × e = 1 modulus phi(n); d is kept as the private key exponent.

Now,

(e, n) constitutes the Public Key

(d, n) constitutes the Private Key

5.2.2.2 Encryption and Decryption Process

Suppose user A wants to send a private message, M, to user B.

i) User A gets User B's public key from some public source.

ii) User A encrypts message M using B's public key. This produces a cipher text message, C using encryption function i.e.
$C = M^e \bmod n$

iii) Cipher text message C is sent over some communication channel

iv) Upon reception, user B decrypts message C using his private key. This produces original plain text message, M using decryption function i.e.
$M = C^d \bmod n$

where,

'C' denotes the cipher text , 'M' denotes the plain text , 'e' denotes the public key exponent, 'd' denotes the private key exponent and 'n' denotes the modulus

5.2.3 RSA Digital Signature Scheme

In the RSA digital signature process, the private key is used to encrypt the message digest. The encrypted message digest becomes the digital signature and is attached to the original data.

A digital signature scheme typically consists of three algorithms:

i) A key generation algorithm that selects a private key uniformly at random from a set of possible private keys. The algorithm outputs the private key and a corresponding public key.

ii) A signing algorithm that produces a signature, when given a message and a private key.

iii) A signature verifying algorithm that either accepts or rejects the message's claim to authenticity, when given a message, public key and a signature.

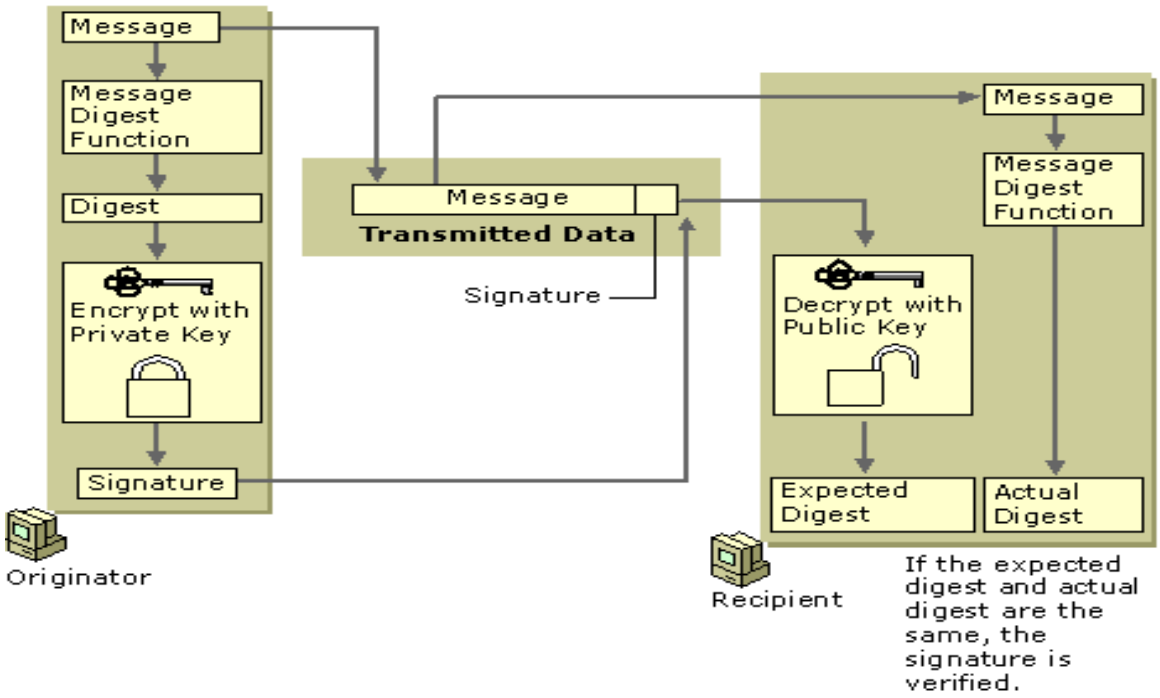

Figure 5.3: RSA Digital Signature Process [36]

5.2.3.1 Verification Process

Sender's End

i) The message digest of the message that is intended to be sent along the channel is calculated using hash function.

ii) The computed message digest is then encrypted using sender's private Key

iii) The signature is appended along with the message and transmitted over the communication channel.

Receiver's End

i) The message digest of the message that is received across the channel is calculated using the hash function.

ii) The digital signature is decrypted using the sender's public key to calculate the expected message digest.

iii) The original message digest is then compared with the expected message digest.

iv) If the original message digest is same as that of expected message digest, document is verified i.e. data is not modified.

5.2.4 ASCII Code Based Encryption Algorithm (ACBEA)

This algorithm is used to encrypt data by using ASCII values of the data to be encrypted. The secret key used to encrypt the data will be modified to another string generating a cipher key that will be used to decrypt the data. The length of the secret key is equal to the length of the plain text to be encrypted.

5.2.4.1 Encryption Process

The steps to encrypt the plain text are as follows [15]:

i) Get the ASCII value of each character of the plain text and store it in an AsciiVal array.

ii) Find out the minimum value (min) from the AsciiVal array.

iii) Now perform modulus operations on each ascii value i.e. AsciiVal[i] and save the result in the ModContent array as follow:

ModContent[i] = AsciiVal[i] % min

If ModContent[i] >=16

Record the positions and go to step 4.

Else

Go to step 5.

iv) Perform modulo 16 operation as

ModContent[i] = ModContent[i] % 16

v) Get the ASCII value of each character of the Secret Key and store it in an AsciiKey array.

vi) Now perform the modulus operation on each ascii value i.e. AsciiKey[i] and save the result in ModKey array as follows:

ModKey[i] = AsciiKey[i] % min

vii) Determine the binary equivalent (each 4 bit) of each element of ModKey and concatenate all values to form a binary string.

viii) Perform the n (length of input) bit right circular shift operation on binary string and determine the encrypt key after the shift i.e. Encrypt[i] by substituting decimal equivalent for each 4 bit binary string.

ix) Add min value to each element of Encrypt[i] to calculate the final cipher key as follows:

CipherKey[i] = Encrypt[i] + min.

x) Now, add each element of ModContent array to the corresponding element of CipherKey array as follows:
Cipher[i] = ModContent[i] + CipherKey[i].

xi) Substitute the ASCII character corresponding to each element of Cipher array to generate the final cipher text.

xii) Substitute the ASCII character corresponding to each element to of the CipherKey array to generate the final cipher key.

5.2.4.2 Decryption Process

The steps for the encryption are as follows [29]:

i) Get the ASCII value of each character of the cipher text and store it in an AsciiCipher array.

ii) Get the ASCII value of each character of the cipher key and store it in an CipherKey array

iii) Perform the subtraction of ascii values of cipher key from cipher text as follows:
Difference[i] = AsciiCipher[i] - CipherKey[i].

iv) Add min value to each value of difference as follow:
DecryptVal[i] = Difference[i] + min

v) Update the DecryptVal by adding 16 to the stored positions.

vi) Substitute the ASCII character corresponding to each element of DecryptVal array to generate the final plain text.

5.2.5 Improved ASCII Code Based Encryption Algorithm (Improved ACBEA)

Improved ASBEA is the enhancement of ACBEA. The main objective behind proposing this algorithm is to overcome the shortcomings of ACBEA and enhance the security of the algorithm at the whole. It uses ASCII values of the data to perform the encryption and decryption process. It can be used to encrypt every possible combination of characters including special symbols, alphabets and digits available in 8-it ASCII character set.

5.2.5.1 Encryption Process

The steps to encrypt the plain text are as follows:-

i) Get the ASCII value of each character of the plain text and store it in an AsciiVal array.

ii) Find out the minimum value (min) from the AsciiVal array.

iii) Now perform the division and modulus operations on each ascii value i.e. AsciiVal[i] and save the result in the DivContent ,ModContent, DivMod array as follows:

DivContent[i] = AsciiVal[i] / min

ModContent[i] = AsciiVal[i] % min

DivMod[i] = ModContent[i] / 16

If ModContent[i] >=16

Go to step 4.

Else

Go to step 5.

iv) Perform modulo 16 operation as

ModContent[i] = ModContent[i] % 16

v) Get the ASCII value of each character of the Secret Key and store it in an AsciiKey array.

vi) Now perform the modulus operation on each ascii value i.e. AsciiKey[i] and save the result in ModKey array as follows:

ModKey[i] = AsciiKey[i] % min

If ModKey[i] >=16

Go to step 7.

Else

Go to step 8.

vii) Perform modulo 16 operation as

ModKey[i] = ModKey[i] % 16

viii) Find out the minimum value (n) from the ModKey array

ix) Determine the binary equivalent (each 4 bit) of each element of ModKey and concatenate all values to form a binary string.

x) Perform the n bit left circular shift operation on binary string and determine the encrypt key after the shift i.e. Encrypt[i] by substituting decimal equivalent for each 4 bit binary string.

xi) Add min value to each element of Encrypt[i] to calculate the final cipher key as follows:

CipherKey[i] = Encrypt[i] + min

xii) Now, add each element of ModContent array to the corresponding element of CipherKey array as follows:

Cipher[i] = ModContent[i] + CipherKey[i]

xiii) Perform binary to gray code conversion on each element of Cipher array, convert the resultant value to corresponding decimal equivalent and update the Cipher array.

xiv) Substitute the ASCII character corresponding to each element of Cipher array to generate the final cipher text.

xv) Substitute the ASCII character corresponding to each element to of the CipherKey array to generate the final cipher key.

5.2.5.2 Decryption Process

The steps for decrypting cipher are as follows:

i) Get the ASCII value of each character of the cipher text and store it in an AsciiCipher array.

ii) Perform gray to binary code conversion on each element of AsciiCipher, convert the resultant value to corresponding decimal equivalent and update the AsciiCipher array.

iii) Get the ASCII value of each character of the cipher key and store it in an CipherKey array

iv) Perform the subtraction of ascii values of cipher key from cipher text as follows:

Difference[i] = AsciiCipher[i] - CipherKey[i].

v) Add min value to each value of difference as follows:

DecryptVal[i] = Difference[i] + min * DivContent[i] + 16 * DivMod[i].

vi) Substitute the ASCII character corresponding to each element of DecryptVal array to generate the final plain text.

5.3 Comparative Analysis of Improved ACBEA over ACBEA

The ACBEA fails to work for every combination of characters (special characters, digits and alphabets) in 8-bit ASCII character set. The limitation of ASBEA algorithm is overcome by Improved ASBEA so that it can encrypt and decrypt every possible combination of ASCII character set. This can be illustrated as below:

Table 5.1: Results of Encryption for ACBEA and Improved ACBEA

	Improved ACBEA					ACBEA			
Plain Text	a	2	!	E		a	2	!	E
Ascii Val	97	50	33	69		97	50	33	69
DivContent	2	1	1	2		-	-	-	-
DivMod	1	1	0	0		-	-	-	-
ModContent	15	1	0	3		15	1	0	3
Secret Key	$	#	u	6		$	#	u	6
AsciiKey	36	35	117	54		36	35	117	54
ModKey	3	2	2	5		3	2	2	5
Binary	0011	0010	0010	0101		0011	0010	0010	0101
Circular Shift	1100	1000	1001	0100		0101	0011	0010	0010
Encrypt	12	8	9	4		5	3	2	2
CipherKey	45	41	42	37		38	36	35	35
Cipher	60	42	42	40		53	37	35	38
Decimal Equivalent of Gray Code	34	63	63	60		-	-	-	-
Final Cipher	“	?	?	<		5	%	#	&
Decryption Key	-	)	*	%		&	$	#	#

Decryption Process

Table 5.2: Results of Decryption for ACBEA and Improved ACBEA

	Improved ACBEA					ACBEA			
Cipher Text	“	?	?	<		5	%	#	&
AsciiCipher	34	63	63	60		53	37	35	38
Decimal Equivalent of Binary Code	60	42	42	40		-	-	-	-
Decryption Key	-	)	*	%		&	$	#	#
CipherKey	45	41	42	37		38	36	35	35
Difference	15	1	0	3		15	1	0	3
Decrypt	97	50	33	69		64	50	33	36
Decrypted Text	a	2	!	E		@	2	!	$

Decryption Successful

Decryption Failed

5.4 Hybrid Cryptosystem

The proposed cryptosystem combines the security of the document by hybrid encryption method and authenticity by digital signatures. A combination of hybrid cryptography and the Digital signatures provides a powerful solution to implement services that guarantee data protection and data integrity. This scheme has all the features of symmetric, asymmetric algorithm and digital signature scheme thereby guaranteeing a more secure environment for data storage and transmission over a network.

5.4.1 Encryption Phase

The various steps involved in the encryption are:

i) First, a secret IDEA key of length 128 bits is generated.

ii) Using this key, the message(M) is encrypted in a quick manner

E_M = IDEA (M)

iii) The IDEA key is encrypted at Level-I using Improved ASCII Code Based Encryption (IACBEA) Algorithm.

E_{K1} = IACBEA (IDEA Key)

iv) Afterwards, the encrypted IDEA key is passed to the RSA Encryption system for the Level-II Encryption

E_{K2} = RSA-Encrypt (E_{K1})

v) Thereafter, the encrypted message (E_M) acts as an input for SHA-512, which will produce 512-bit condensed version.

E_H = SHA-512(EM)

vi) The message digest (E_H) will be signed using RSA Digital Signature algorithm using the private key of the sender, hence generating a digital signature.

D=RSA-Sign (E_H)

vii) The encrypted message (E_M) encrypted IDEA key (E_{K2}) and digital Signature (D) are transmitted across the communication channel.

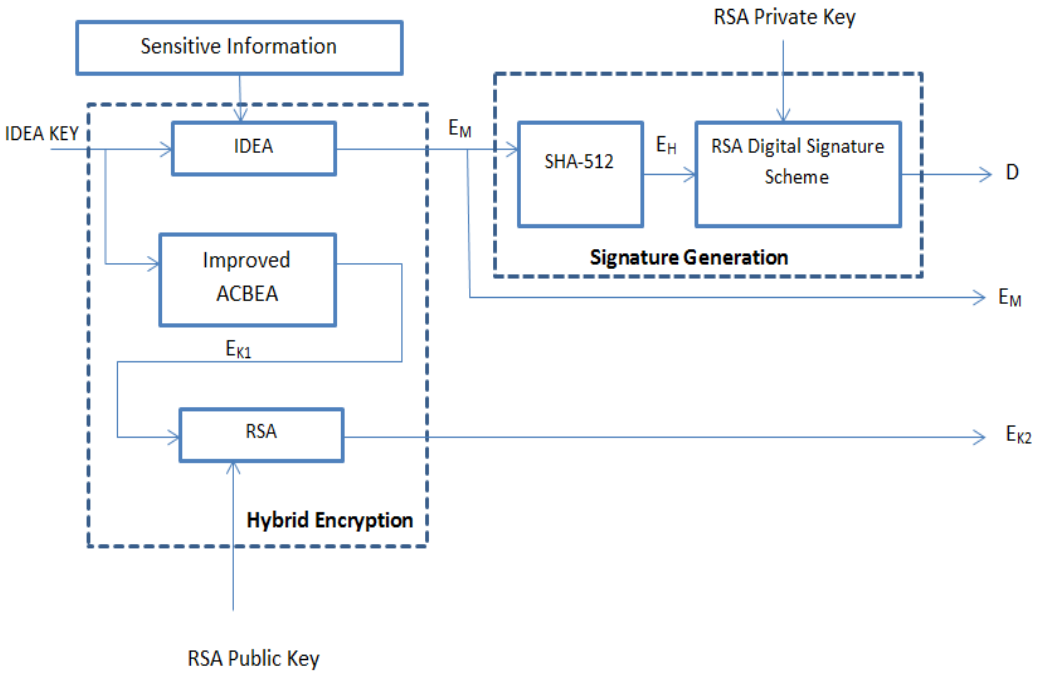

Figure 5.4: Encryption Phase

5.4.2 Decryption Phase

The various steps for the decryption are:

i) The encrypted IDEA key (E_{K2}) is passed to RSA Decryption system for Level-I Decryption

E_{K1} = RSA-Decrypt (E_{K2})

ii) The Level-I decrypted IDEA key (E_{K1}) is decrypted using Improved ASCII Code Based Encryption Algorithm.

K = IACBEA (E_{K1})

iii) The original message (M) is retained by IDEA algorithm using the obtained IDEA Key (K).

M=IDEA (E_M)

iv) The encrypted message (E_M) acts as an input for SHA-512, which produces 512-bit condensed version (E_H)

E_H = SHA-512 (E_M)

v) The digital signature acts as an input to RSA Digital Signature Verification algorithm, which produces the expected hash (E_H) using the sender's public key.

E_H = RSA-Verify (D)

vi) In order to verify the origin and the integrity of data, the calculated hash (E_H) and expected hash (E_H) are compared.

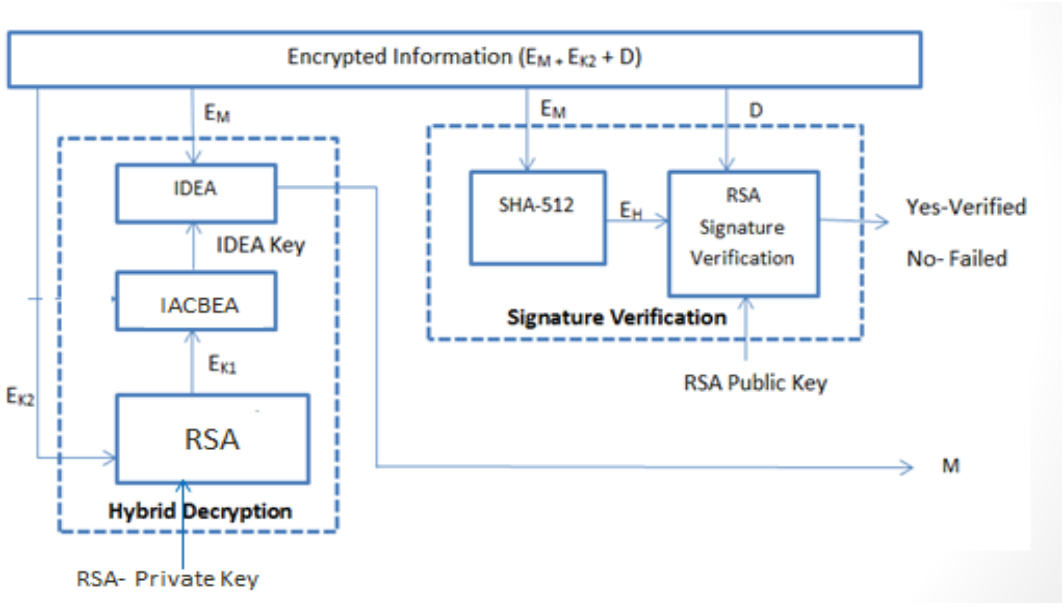

Figure 5.5: Decryption Phase

5.5 Security Architecture for Cloud Storage

The new trend of outsourcing data on the cloud; where users have reduced control of their own information, has given rise to many security threats. There is an emerging need to provide new security architecture for customers to ensure that the data stored in the cloud is protected against security threats. As the remote cloud server cannot be fully trusted, so one of the security measures is to encrypt the data before outsourcing. Thus, to ensure the security of data stored on the remote cloud servers, the proposed hybrid cryptographic scheme is employed.

The scheme can be broadly categorized into three phases:

i) Authentication Phase
ii) Uploading Phase
iii) Downloading Phase

The cloud environment is realized using OpenNebula toolkit. OpenNebula consists of one front node/server and n cluster nodes. The VM's deployed form front node to the cluster node, serve as a distributed remote servers providing Data as a service.

5.5.1 Authentication Phase

All the users, irrespective of new or existing member, need to pass through a secured channel which is connected to the main server (front node). The authentication is required to keep the user account secure and secret from the unauthorized user. When a user needs to use the cloud services i.e. download or upload a file from or to the remote server, he needs to authenticate himself first before proceeding any further.

The steps involved in the authentication phase are:

i) The system will accept the user name and password from the user.
ii) If entered name and password are valid, the system will establish a connection with the cloud; otherwise, the system shows an error and rejects the user.

The next steps work on the condition that the user is authenticated and a connection with the cloud has been established. As a result only authorized user with a valid account will be able to connect to the cloud system.

5.5.2 Uploading Phase

Various steps involved in the uploading phase are listed as below:

i) Client sends request to main server to authenticate himself.

ii) On the successful authentication, a secured connection is established with the cloud.

iii) Once the connection is established, the user encrypts the file using proposed hybrid cryptographic scheme.

iv) After the files are encrypted, client provides a unique ID and sends a request to main server to upload the files.

v) In response to the client's request, the server returns the IP address of VM having the minimum load among the available VM's on the network.

vi) Encrypted file(s) are uploaded to the VM server.

vii) Client disconnects the connection with the cloud, and the load on the VM is updated.

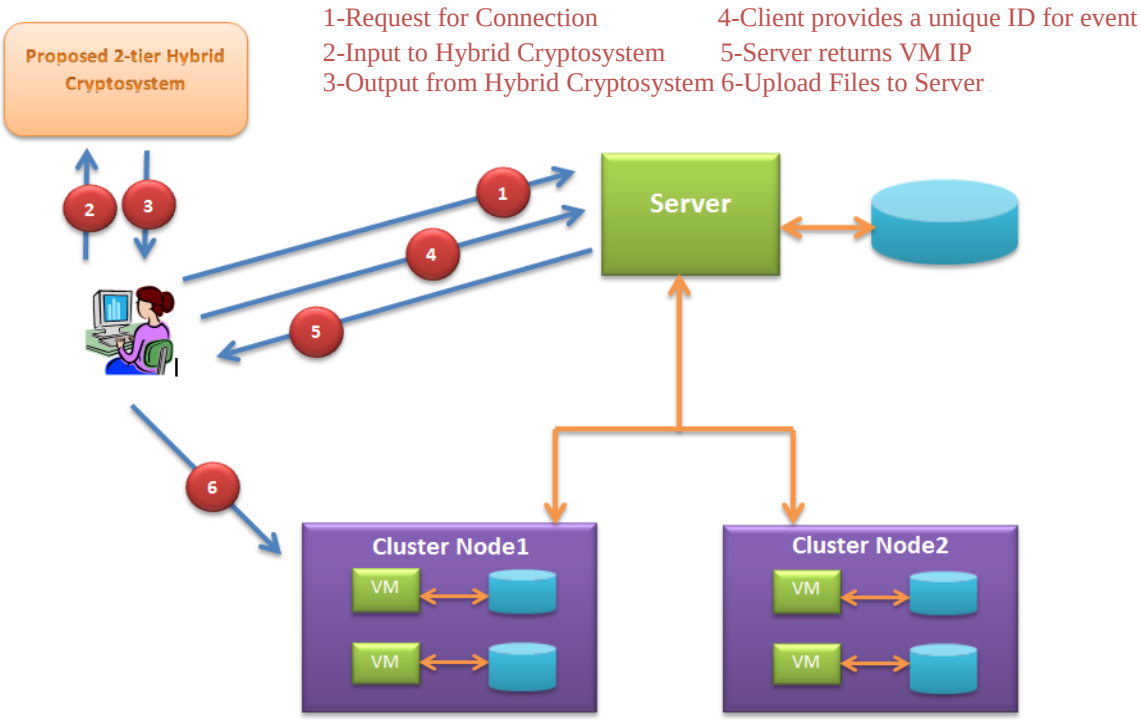

Figure 5.6: Uploading Phase

5.5.3 Downloading Phase

Various steps involved in the authentication phase are listed below:

i) Client sends a request to main server to authenticate himself.

ii) On successful authentication, a secured connection is established with the cloud.

iii) Client is required to enter the unique ID which he had entered during the uploading of the file(s).

iv) Main server returns the IP address of VM corresponding to the unique ID.

v) Encrypted files are downloaded from the VM server whose IP address is returned from the main server.

vi) Decryption is carried out at the client end using the proposed hybrid cryptographic scheme.

vii) Client disconnects the connection established with the cloud and the load on the VM is updated.

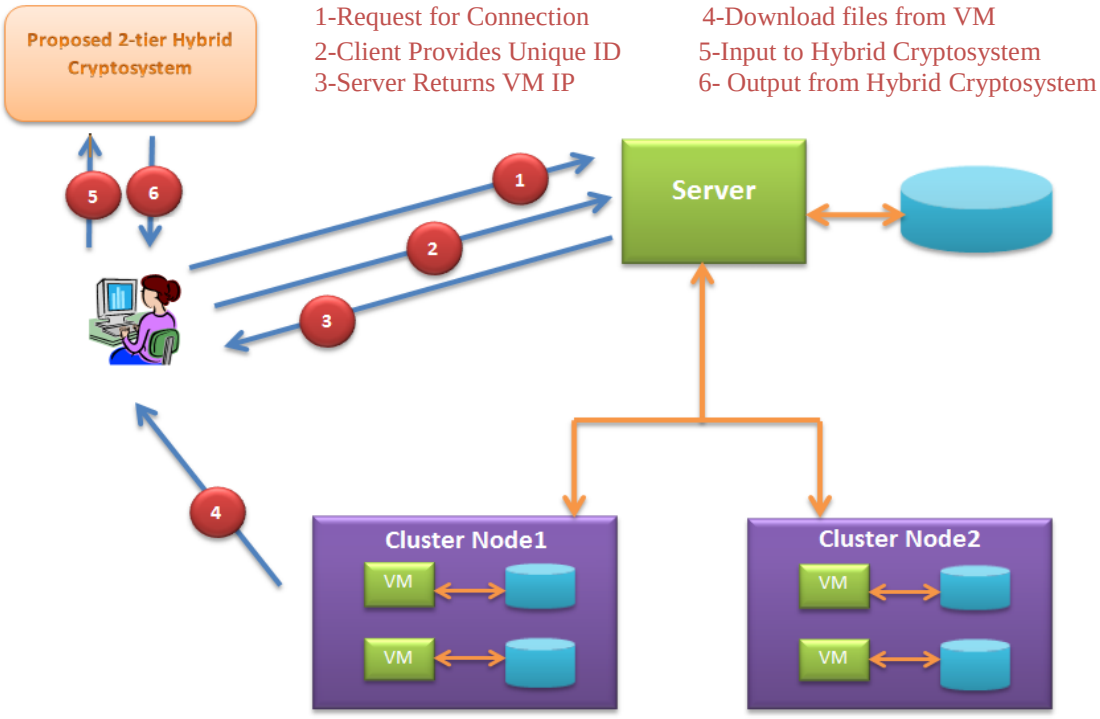

Figure 5.7: Downloading Phase

Chapter 6

Results

6.1 Authentication Phase

6.1.1 Login Frame

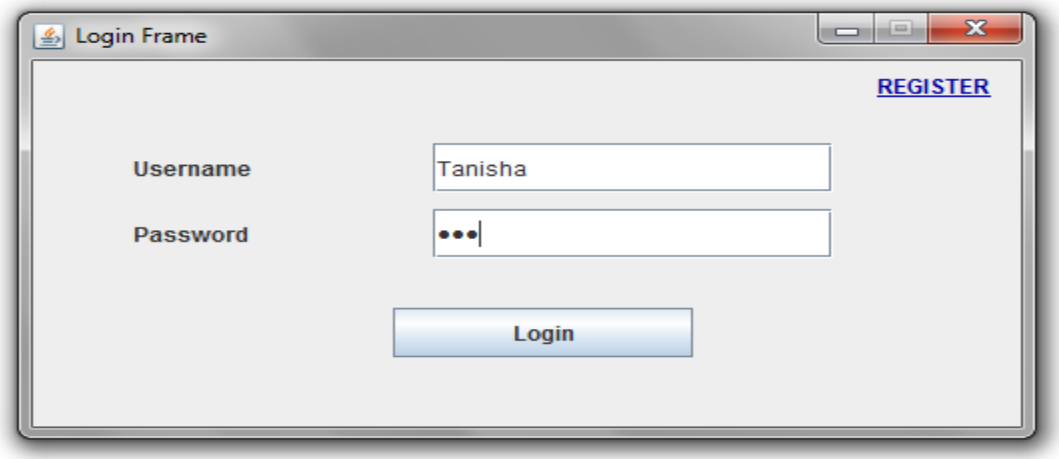

Figure 6.1: Login Frame

Registered user will enter the corresponding login credentials into the Login frame. The main server authenticates the user details; if the details are verified, a connection is established with the remote server and user is directed to the next page. If details are not verified, the user cannot proceed further and is notified accordingly.

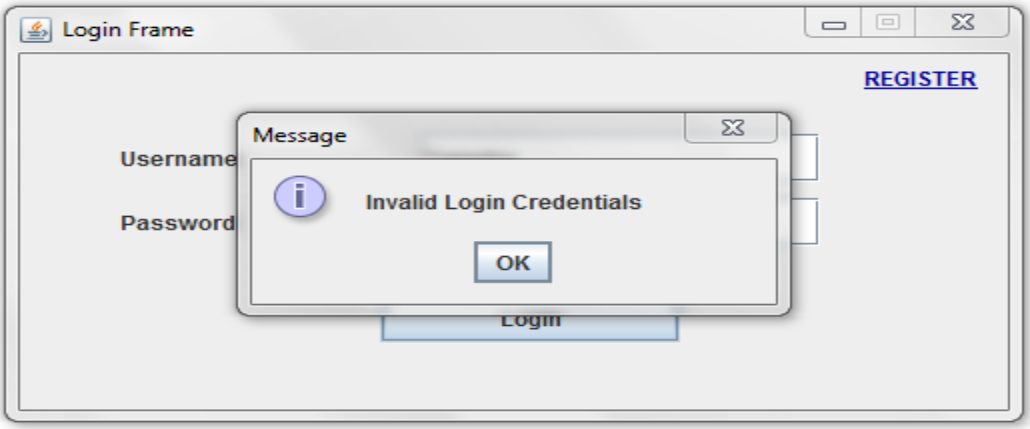

Figure 6.2: Login Failed

6.1.2 Registration Frame

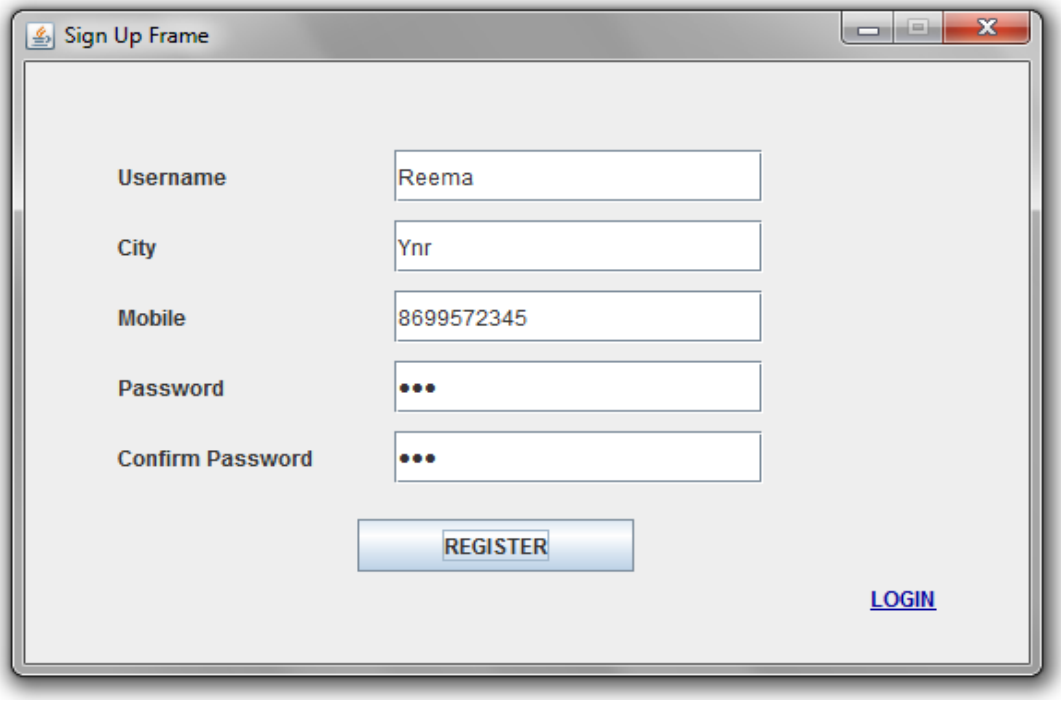

Figure 6.3: Registration Frame

First time users can register themselves by entering their corresponding details into the Registration form. Once the user is successfully registered to the server's database, the user can login and perform the encryption/decryption operations.

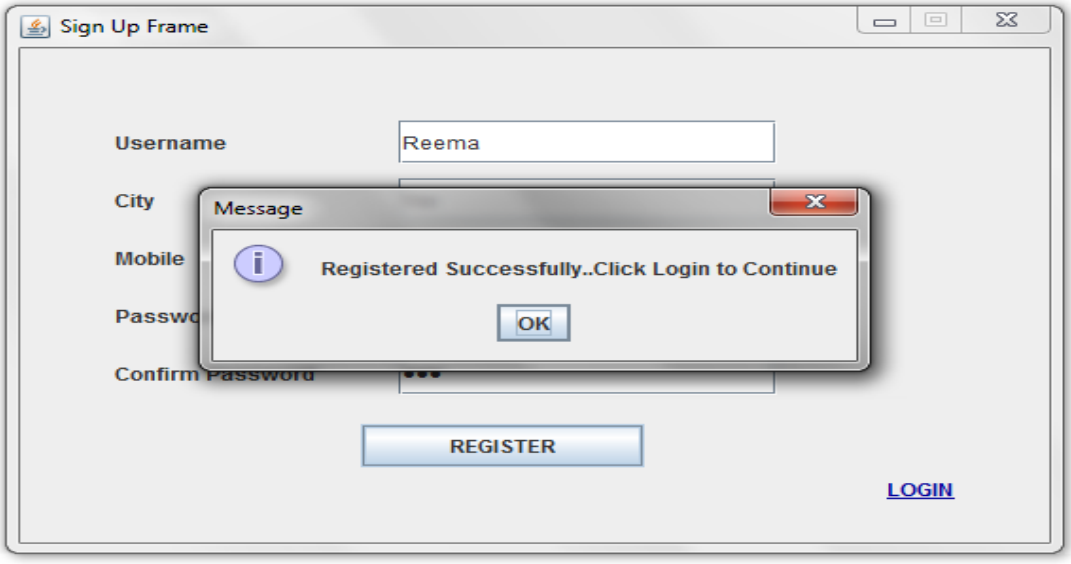

Figure 6.4: Successful Registration

6.2 Operation Selection Phase

After successful login, user will be directed to the Operation Selection frame. The user will select the required operation and proceed further by clicking NEXT.

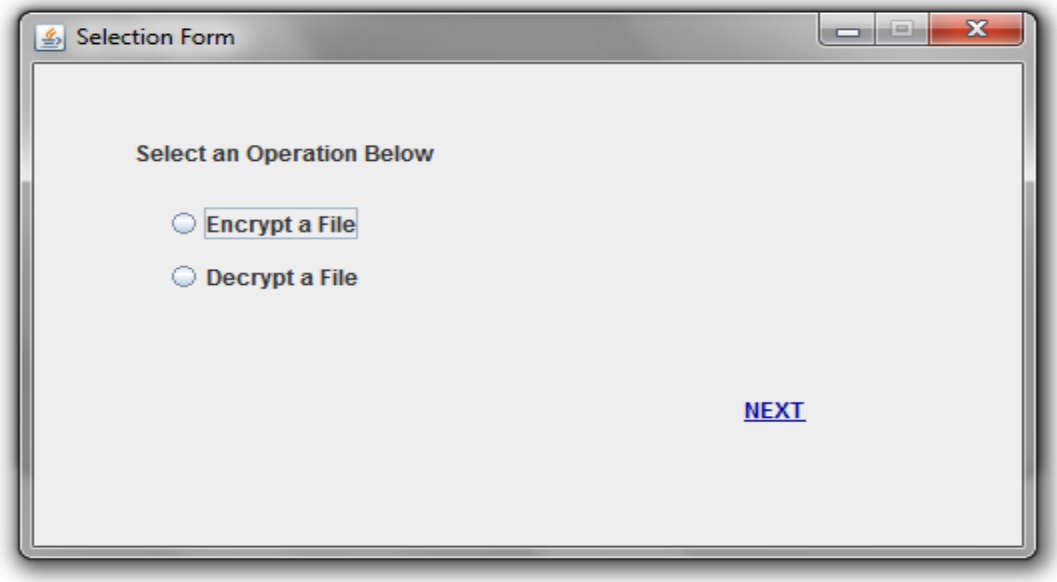

Figure 6.5: Selection Frame

If user clicks NEXT without specifying any of the operation, the user cannot proceed further and will be notified accordingly.

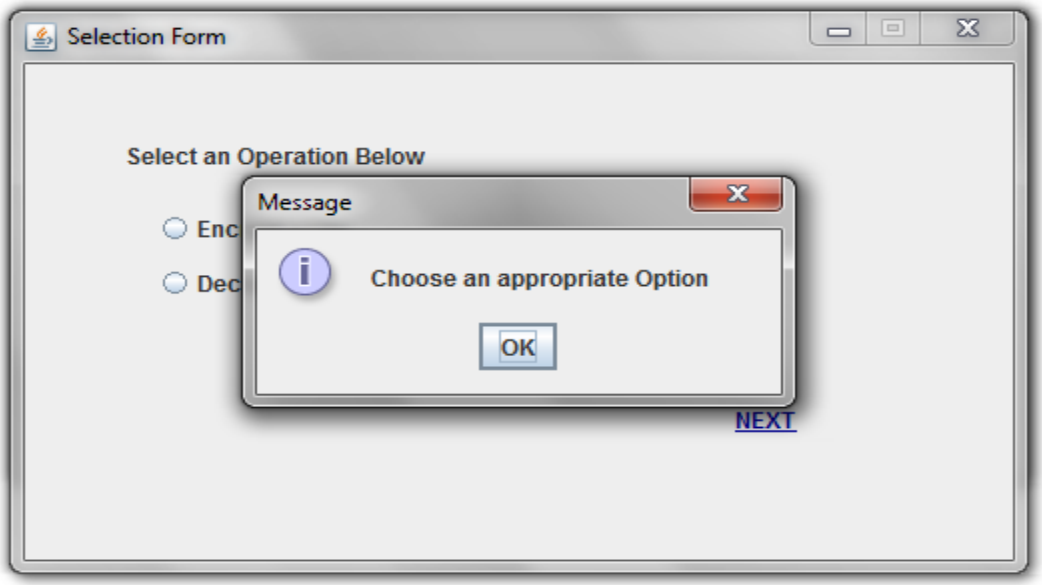

Figure 6.6: Client End Validation

6.3 Encryption Phase

6.3.1 Encryption Frame

If the user has selected option for encrypting file, the user is directed to Encryption frame where the user is required to browse the target file and specify the 128 bit IDEA key.

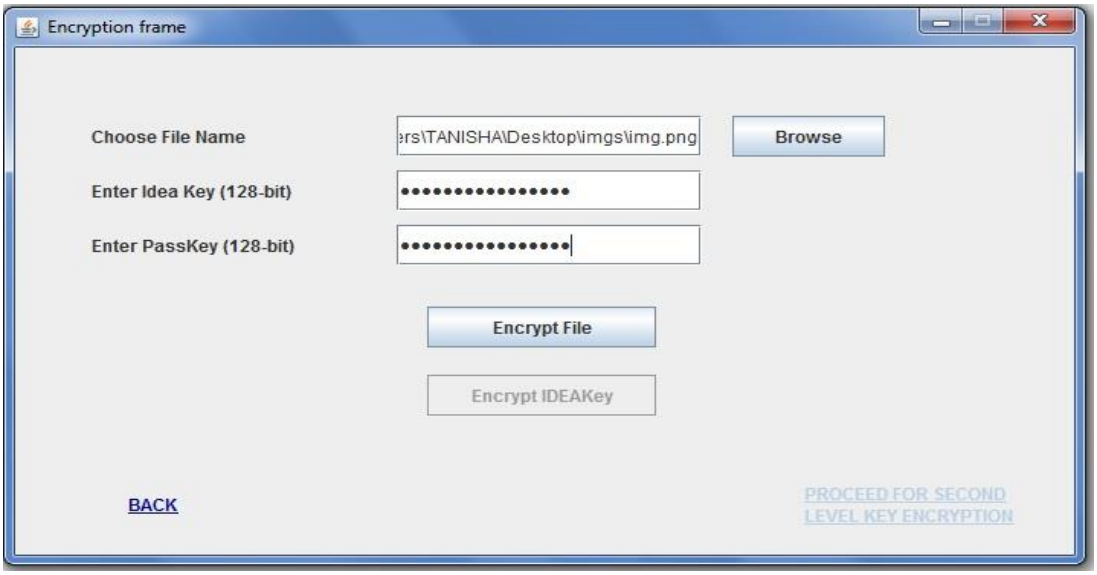

Figure 6.7: Encryption Frame

If the target file is encrypted successfully, user is notified and thereafter the user can proceed for Level-I encryption of the IDEA key.

Figure 6.8: Encryption Successful

Figure 6.9: Level-I Key Encryption

The IDEA key used to encrypt the file is encrypted using Improved ASCII Code Based Encryption Algorithm. Encryption process is carried out using 128 bit pass key which is provided by the user at the time of encryption. If the encryption process is successful, client is notified accordingly

.

6.3.2 Level-II Key Encryption

After, IDEA key is encrypted at Level-I, it is passed to RSA encryption system where the output of Level-I encryption is encrypted using the RSA public key. The output is 128 bit encrypted IDEA key.

Figure 6.10: Level-II Key Encryption

6.3.3 Signature Generation Frame

Once, Level-I and Level-II encryption of IDEA key is successfully completed, the user will proceed for creating digital signature of the encrypted file. If the digital signature is generated successfully, the user will further upload the files to the server by clicking PROCEED FOR UPLOADING button on the frame.

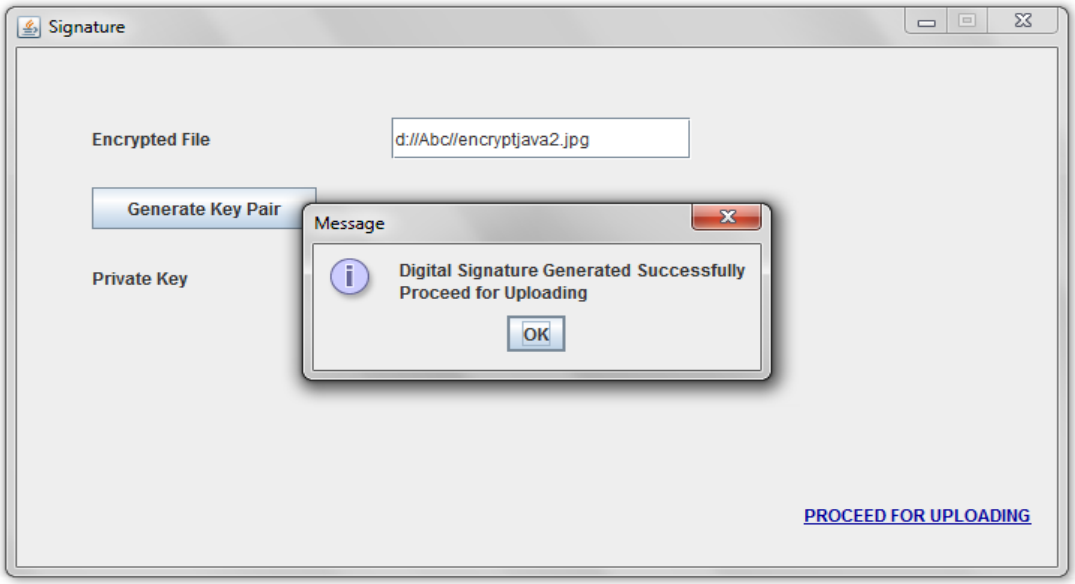

Figure 6.11: Digital Signature Generated

6.3.4 Upload Frame

After creating the digital signature of the encrypted file, the user is required to upload encrypted file, encrypted IDEA key as well as digital signature to the remote server. The user needs to specify a unique id through which user can get access over the uploaded files whenever required. When the files are successfully uploaded at server end, the user is notified accordingly.

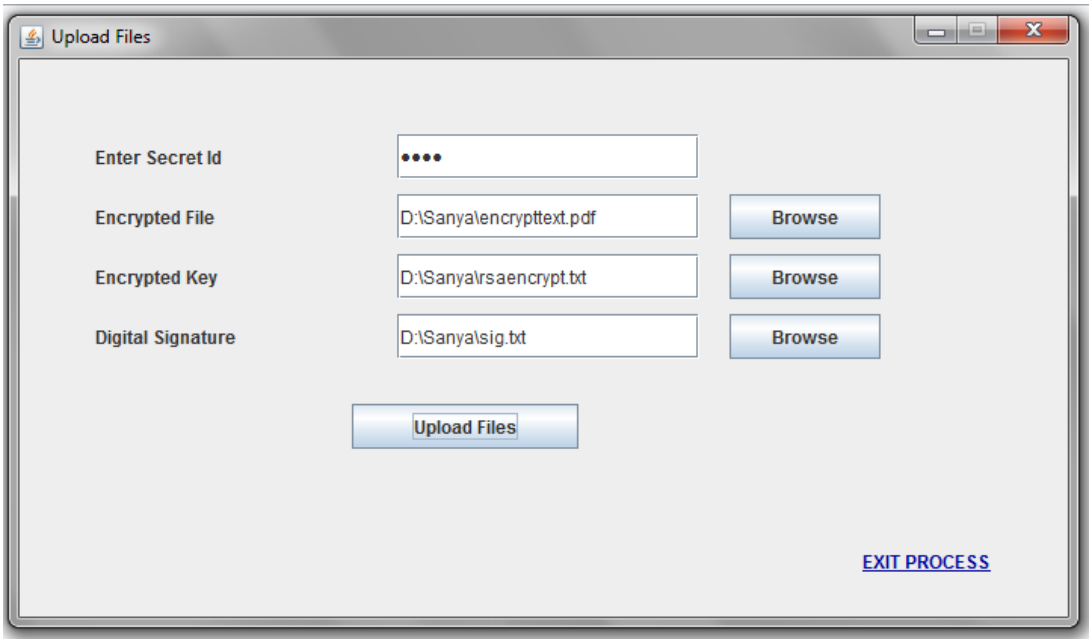

Figure 6.12: Upload Frame

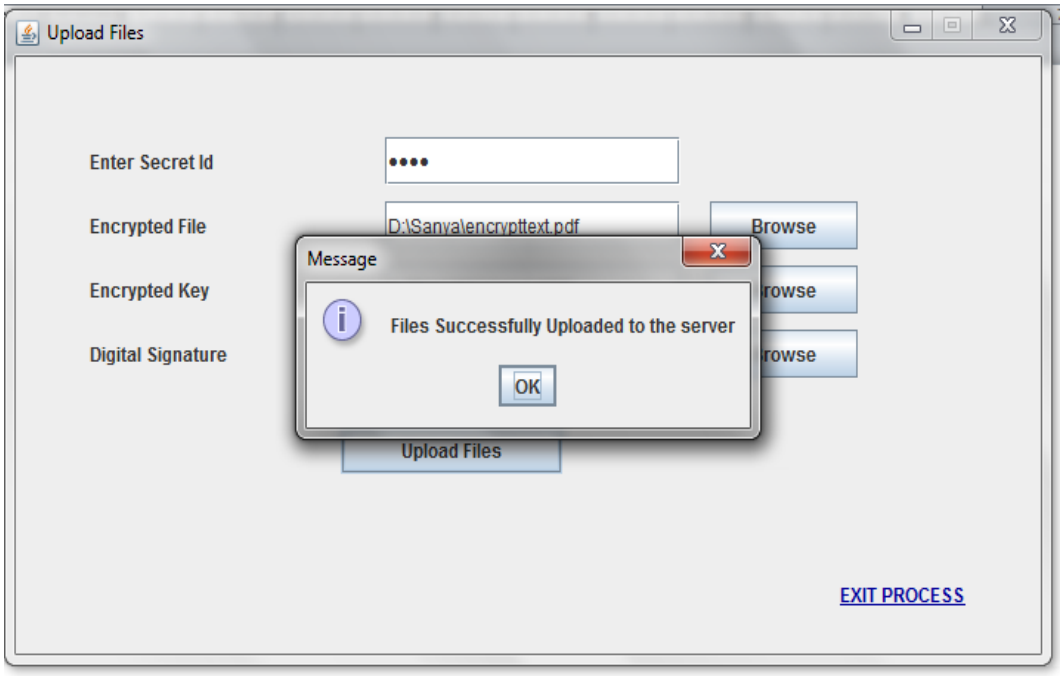

Figure 6.13: Uploading Successful

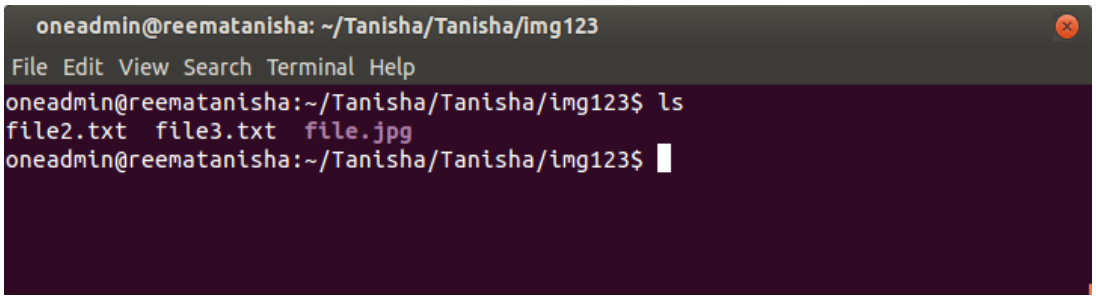

Figure 6.14: Uploaded Files at Server End

6.4 Decryption Phase

6.4.1 Download Frame

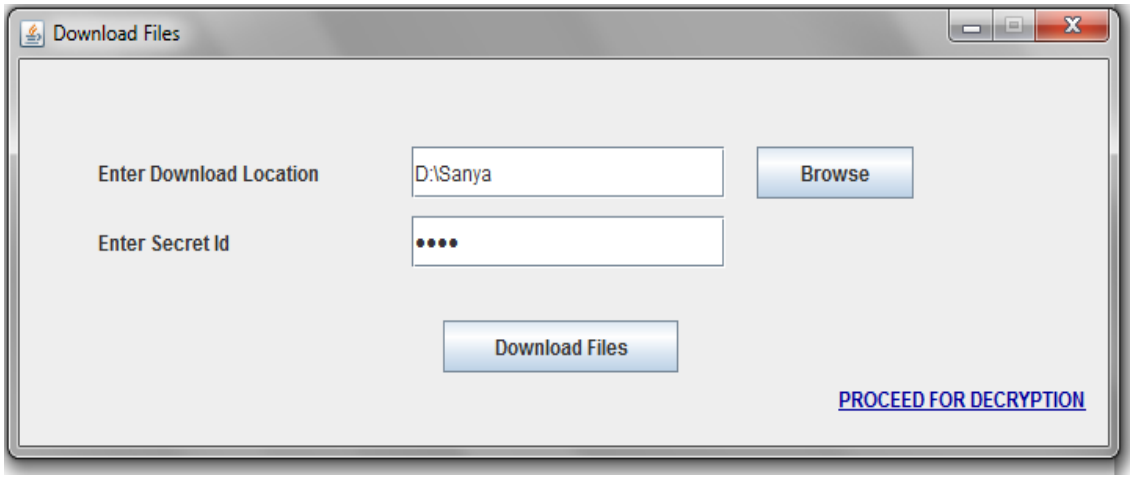

Figure 6.15: Download Frame

If the user selects the decryption operation; first of all, the user is directed to download frame where the user needs to specify the download location and the unique id corresponding to the uploaded files. On clicking the Download Files button, the files are downloaded at the specified location. The downloaded files include the encrypted file, encrypted IDEA key and the Digital signature. Once, the files are downloaded at user's end, the user can begin the decryption process by clicking on PROCEED FOR DECRYPTION button on the Decryption frame.

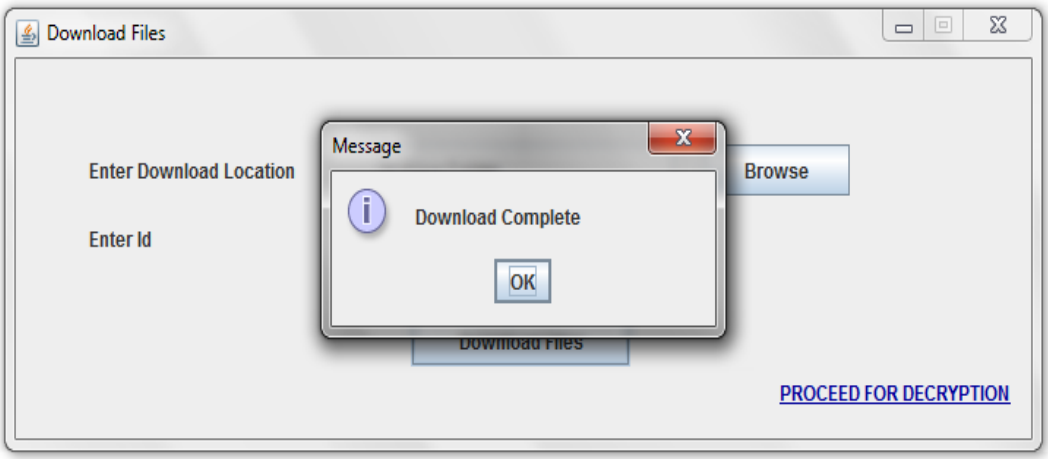

Figure 6.16: Download Complete

6.4.2 Decryption Frame

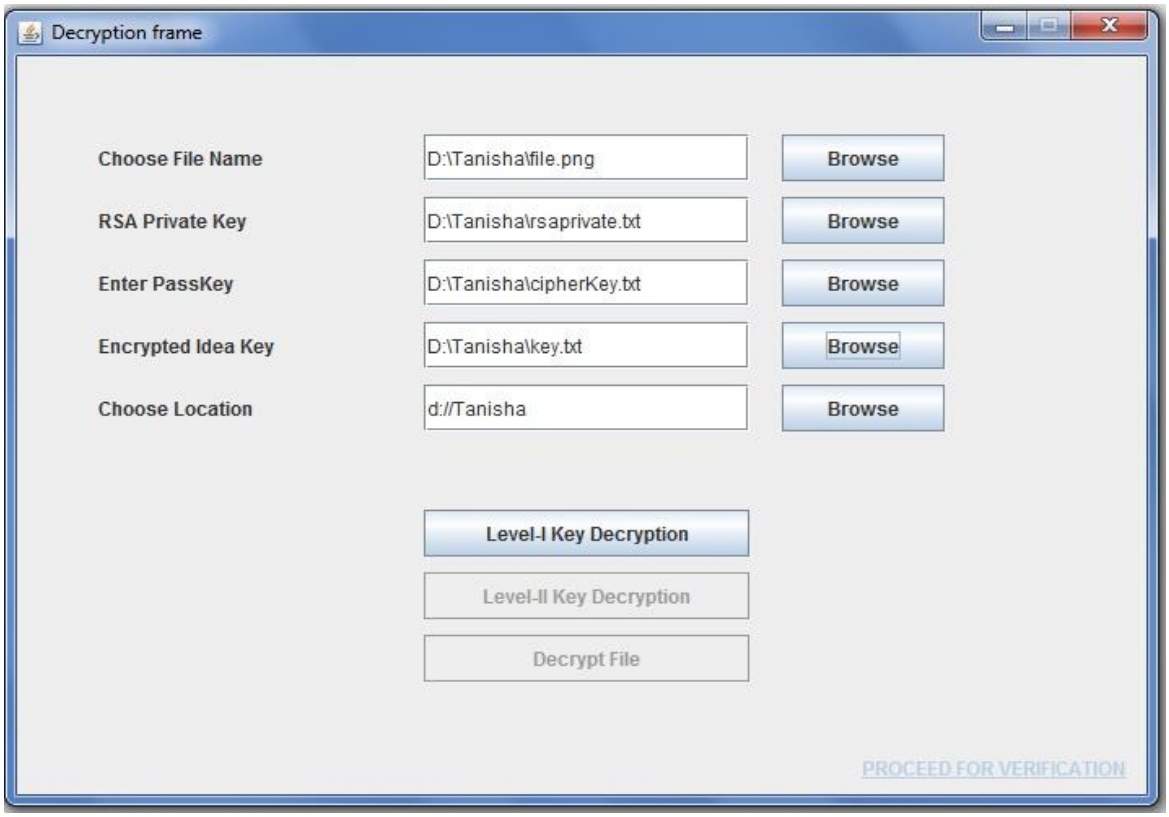

Figure 6.17: Decryption Frame

After downloading the files, the user will proceed for decrypting the downloaded file. In the decryption frame user needs to specify the location of downloaded encrypted file as well as encrypted IDEA key. The user is also required to provide

the Improved ACBEA Passkey and RSA private key for Level-I and Level-II decryption of IDEA key respectively

The decryption is carried out in three steps:

i) Encrypted IDEA key is decrypted using RSA private key (Level-I Decryption).

ii) The output yielded after the RSA decryption is passed to Improved ACBEA along with the pass key for obtaining the original IDEA key (Level-II Decryption).

iii) Decrypted IDEA key is used to decrypt the encrypted file.

After Decrypting the file, the data integrity and origin of file is verified by the Digital Signature Verification Process.

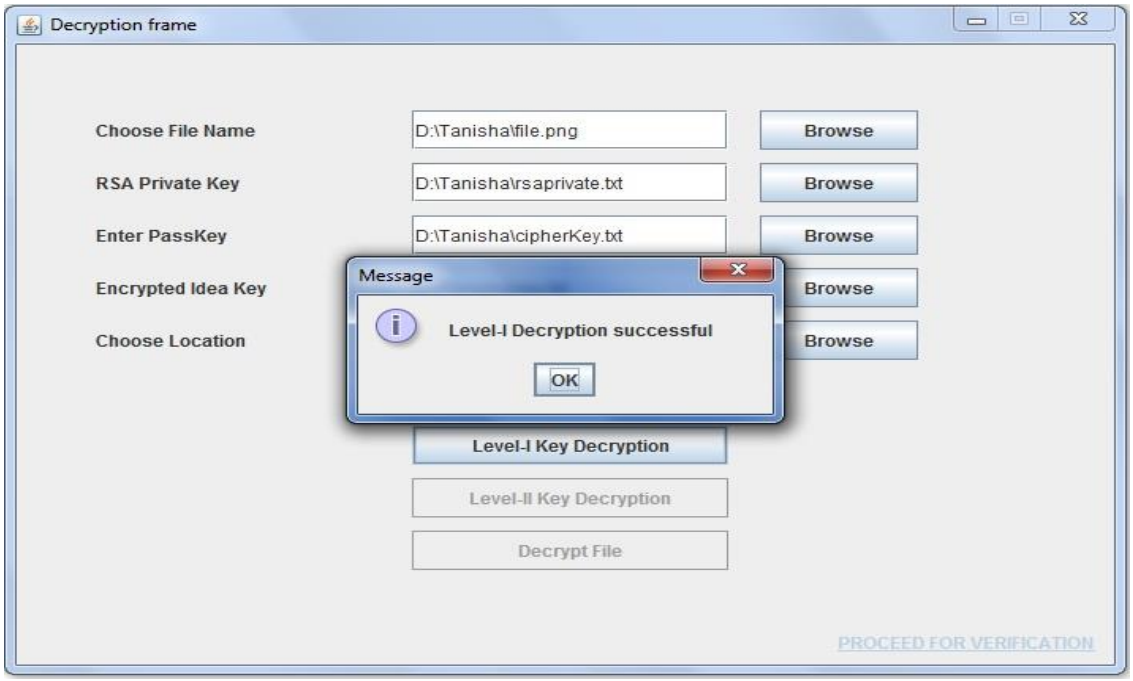

Figure 6.18: Level-I Decryption

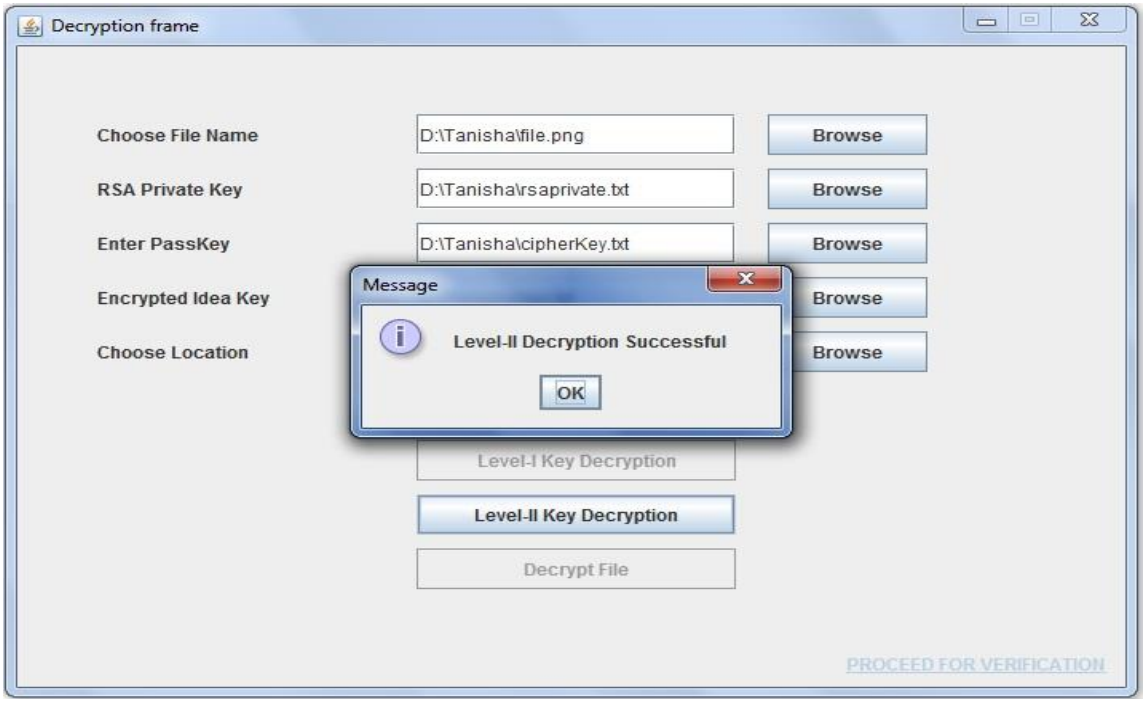

Figure 6.19: Level-II Decryption

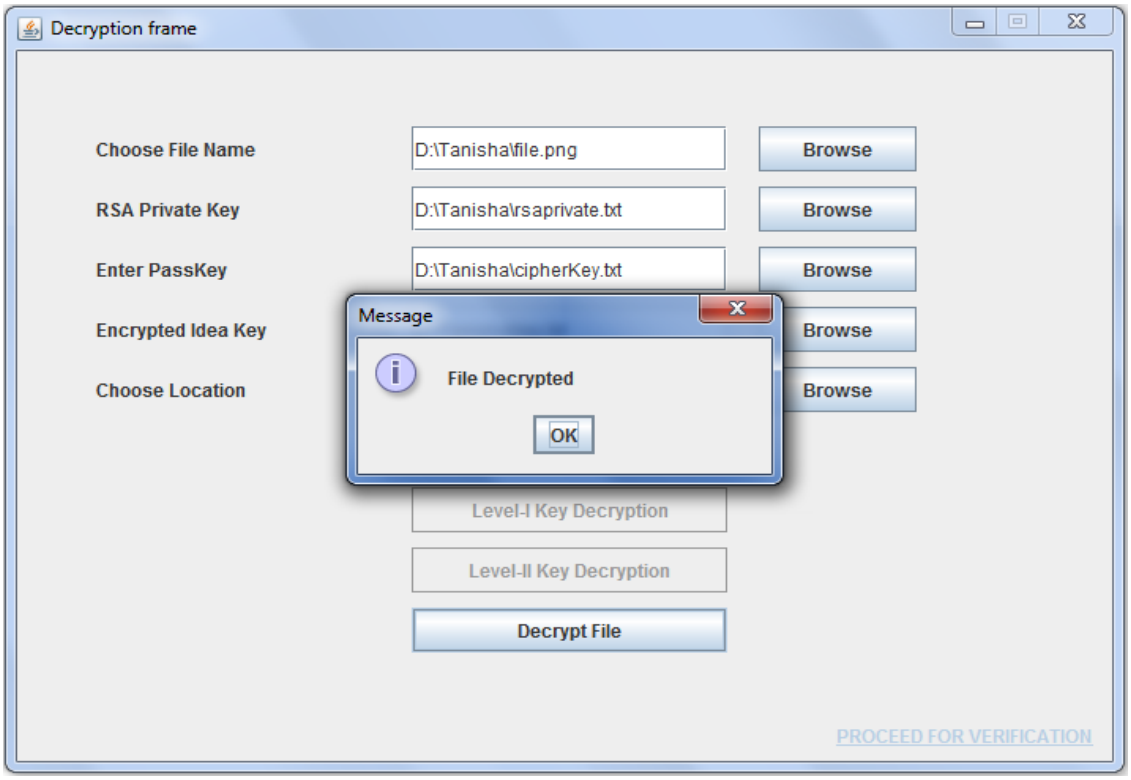

Figure 6.20: File Decrypted Successfully

Figure 6.21: Decrypted File

6.4.3 Signature Verification Frame

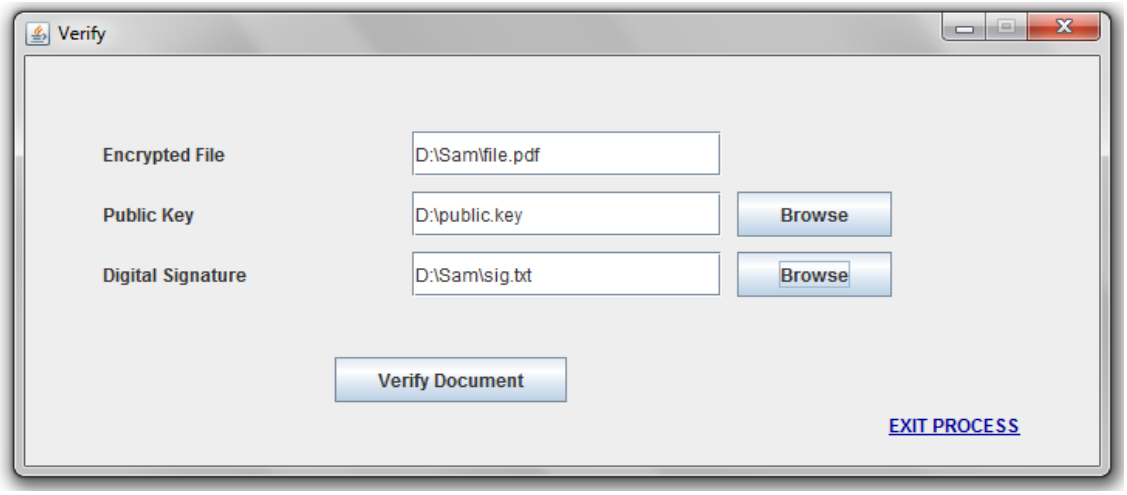

Figure 6.22: Verification Frame

After decrypting the file, the user is directed to the Signature Verification frame. Digital Signature Verification ensures the data integrity and the origin of the file. The user is required to enter the public key and the digital signature obtained from the server. The public key is used to obtain the expected message digest by decrypting the digital signature. This hash is compared against the actual message digest computed from received encrypted file. If both hashes are found out to be

same, the verification succeeds otherwise fails. The user is notified in both the cases.

Verification Successful

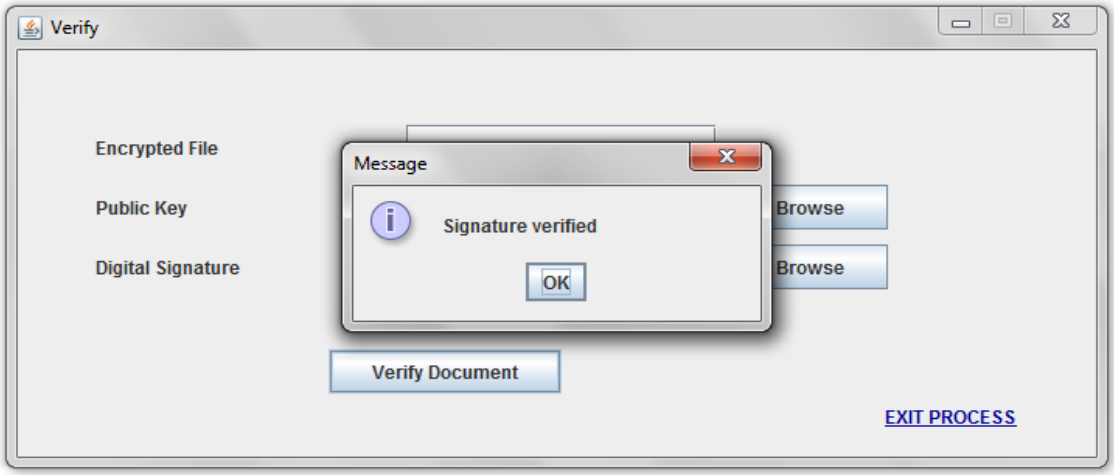

Figure 6.23: Verification Successful

Verification Failure

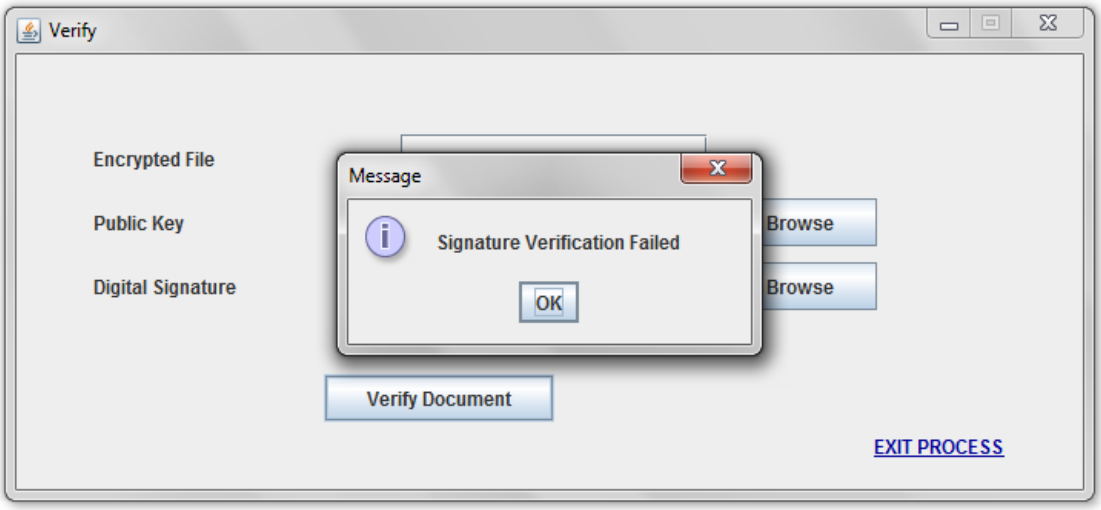

Figure 6.24: Verification Failure

After the document verification, the user can exit process by clicking EXIT PROCESS button.

6.4.4 Logout Frame

Once the process of encryption or decryption is complete, user finally disconnects the connection established with the cloud and is redirected to the logout frame.

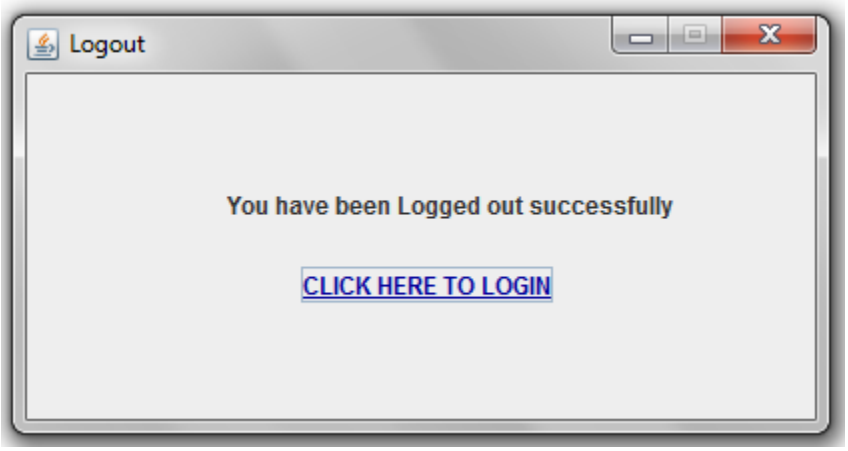

Figure 6.25: Logout Frame

Chapter 7

Conclusion and Future Scope

This chapter discusses conclusion of the work presented in this thesis. The chapter ends with discussion of the future direction which thesis work can take.

7.1 Conclusion

The cost of maintaining the data center is increasing rapidly, especially for the medium data centers. An economic choice is to use cloud computing and cloud storage instead of managing data center by itself. Cloud computing moves the databases to the large data centers, where the management of the data and services may not be fully trustworthy. Since, the outsourced data resides in provider premises; there arises difficulty of ensuring that data is safe in cloud storage. This problem can be solved by encrypting the data before outsourcing, using efficient cryptographic algorithms.

The proposed methodology suggests a hybrid cryptographic scheme coupled with the digital signature scheme for encryption of the files to be uploaded on the cloud. A combination of hybrid cryptography and the Digital signatures provides a powerful solution to implement services that guarantee data protection and data integrity. Secondly, the two tier encryption of symmetric key further adds to the security. IDEA algorithm in encryption/decryption process shows high efficiency and ease of implementation. Also, IDEA uses 128 bit key that is strong enough against various cryptographic attacks. In fact, there are no linear cryptanalytic attacks on IDEA, and there are no known algebraic weaknesses in IDEA. RSA algorithm and Improved ACBEA, used to encrypt the symmetric key, ensures the safe delivery of symmetric key necessary for encrypting/decrypting data. RSA Digital Signature Scheme ensures authenticity and integrity of data.

7.2 Future Scope

Future opportunities could explore following:

i) The proposed framework has been implemented on image, word, text and pdf files. This can be further enhanced to encrypt audio and video files.

ii) The IDEA algorithm has a few classes of weak keys, given its present key schedule mechanism. Key scheduling mechanism can be improved or redesigned in the near future to eliminate classes of weak keys for the IDEA block cipher algorithm.

iii) IDEA uses a 64 bit block for encryption/decryption process. The greater will be size of block, the greater will be the security. Thus, a detailed analysis and work needs to be done so that block size in IDEA can be improved from 64 bit to ensure the maximum range of protection against attacks.

References

[1] R. Buyya *et al.,* "Introduction to Cloud Computing", *Cloud Computing Principles and Paradigms,* USA: John Wiley & Sons, 2011.

[2] *http://www.kloudpros.com/primer-cloud-computing/types-of-cloud-deployments.*

[3] Sun Microsystems, "Introduction to Cloud Architecture", *http://eresearch.wiki. otago.ac.nz/images/7/75/Cloudcomputing.pdf,* June 2009.

[4] B. Rimal *et al.,* "A Taxonomy and Survey of Cloud Computing Systems", *Proc. 5th International Joint Conference on INC, IMS and IDC,* pp: 44-51, 2009.

[5] *http://acloudyplace.com/2011/a-comprehensive-introduction-to-cloudcomputing/.*

[6] *http://searchcloudcomputing .tech target.com/definition/.*

[7] G. Laszewski *et al.,* "Comparison of Multiple Cloud Frameworks", *Proc. 5th IEEE International Conference on Cloud Computing*, pp: 734-744, June 2012.

[8] P. Sempolinski et *al.,* "A Comparison and Critique of Eucalyptus, OpenNebula and Nimbus", *Proc. 2nd IEEE International Conference on Cloud Computing Technology and Science*, pp: 417-426, 2010.

[9] X. Wen *et al., "*Comparison of Open-Source Cloud Management Platforms: OpenStack and OpenNebula", *Proc. 9th IEEE International Conference on Fuzzy Systems and Knowledge Discovery,* pp: 2457-2461, 2012.

[10] H. Tianfield, "Security Issues in Cloud Computing", *Proc. IEEE International Conference on Systems, Man, and Cybernetics*, October 14-17, 2012.

[11] A. Behl *et al.*, "An Analysis of Cloud Computing Security Issues", *Proc. IEEE Conference on World Information and Communication Technologies*, pp: 109-114, 2012.

[12] P. Subhasri, "Cloud Computing: Security Challenges & Encryption Practices", *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, March 2013.

[13] Z. Tang *et al.*, Study on Data Security of Cloud Computing, *Proc. IEEE Spring Congress on Engineering and Technology,* pp: 1-3, 2012.

[14] I. Basharat *et al.*, "Database Security and Encryption: A Survey Study", *International Journal of Computer Applications,* vol. 47, no.12, pp:28-34, June 2012.

[15] A. Mathur , " A Research paper: An ASCII value based data encryption algorithm and its comparison with other symmetric data encryption algorithms", *International Journal on Computer Science and Engineering* , vol. 4, no. 9, pp:1650- 1657, September 2012.

[16] *http://mscerts.program ming4.us/programming/cloud-security-and-privacy.*

[17] *http://en.kryptotel.net/encryption.html*.

[18] *http://www.cryptographyworld.com/key.htm*.

[19] A. Pillai *et al.,* "A Study on Open Source Cloud Computing Platforms", *International Journal of Multidisciplinary Management Studies,* vol.2, issue 7, July 2012.

[20] C. Yang *et al.,* "On Construction of Cloud IaaS Using KVM and OpenNebula for Video Services", *Proc. 41st IEEE International Conference on Parallel Processing Workshops*, pp*:* 212-221, 2012.

[21] R. Wadhawan, " Cloud Computing Security Issues in Infrastructure as a Service", *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 2, issue 1, January 2012.

[22] D. Chen *et al.*, "Data Security and Privacy Protection Issues in Cloud Computing", *Proc. IEEE International Conference on Computer Science and Electronics Engineering*, vol. 1, pp: 647-651, 2012.

[23] K. Nafi, "A Newer User Authentication, File encryption and Distributed Server Based Cloud Computing security architecture", *International Journal of Advanced Computer Science and Applications,* vol. 3, no. 10, 2012.

[24] D. Mukhopadhyay, "Enhanced Security for Cloud Storage using File Encryption", *Proc. International Conference on Distributed Computing and Internet Technologies*, 2013.

[25] P. Rewagad *et al.,* "Use of Digital Signature with Diffie Hellman Key Exchange and AES Encryption Algorithm to Enhance Data Security in Cloud Computing",

Proc. *IEEE International Conference on Communication Systems and Network Technologies*, pp: 437-439, 2013.

[26] U. Somani *et al.*, "Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing", *Proc. 1st International Conference on Parallel, Distributed and Grid Computing,* pp: 211-216, 2010.

[27] A. Mohta *et al.,* "Robust Data Security for Cloud while using Third Party Auditor", *International Journal of Advanced Research in Computer Science and Software Engineering,* vol.2 , issue 2, February 2012.

[28] Y.P. Singh *et al., "*On the security of Joint Signature and Hybrid Encryption", *Proc. 13th IEEE International Conference on Networks,* vol. 1, 2005.

[29] G. Toraldo, "OpenNebula 3 Cloud Computing", *OpenNebula and Why it Matters,* UK: PACKT, May 2012.

[30] T. Cordeiro *et al.*, "Open Source Cloud Computing Platforms", *Proc. 9th International Conference on Grid and Cloud Computing,* pp: 366-371, 2010.

[31] *http://opennebula.org/.*

[32] S. Basu, "International Data Encryption Algorithm (IDEA) – A Typical Illustration", *Journal of Global Research in Computer Science* , vol. 2, no. 7, pp: 116-118, July 2011.

[33] N.Hoffman, "A Simplified IDEA Algorithm", *Journal Cryptologia,* vol. 31, issue 2, pp: 143-151, April 2007.

[34] *http://en.wikipedia.org/wiki/ International_ Data _Encryption_Algorithm.*

[35] J. Yadav *et al*., "Modified RSA Public Key Cryptosystem Using Short Range Natural Number Algorithm", *International Journal of Advanced Research in Computer Science and Software Engineering* , vol. 2, August 2012.

[36] *http://www.eeweb.com/project/shanthanmudhasani/digital-signature-algorithm.*