A Project Report
on

ENHANCED SECURITY IN CLOUD COMPUTING

Submitted in partial fulfillment of the requirements
for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering

by
Abhishek Verma
1613101866
17SCSE101175

Semester VIII

Under the Supervision of
Mr. Ashok Kumar Yadav

GALGOTIAS
UNIVERSITY

TABLE OF CONTENTS

Title

CERTIFICATE

This is to certify that the project work entitled **ENHANCED SECURITY IN CLOUD COMPUTING** Is a bonafide work carried out by **Abhishek Verma (1613101866)** is partial fulfillment of the requirements for the award of degree of BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE ENGINEERING by GALGOTIAS UNIVERSITY**,**Greater noida under our guidance supervision.

The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.

Mr.Ashok Kumar

STUDENT'S DECLARATION

I hereby declare that the work being presented in this report entitled **"ENHANCED SECURITY IN CLOUD COMPUTING"** is an authentic record of my own work carried out under the supervision of Mr.Ashok Kumar **.**

The matter embodied in this report has not been submitted by me for the award of any other degree.

Dated :Signature of student **(ABHISHEK VERMA)**

Department:SCSE

ACKNOWLEDGEMENT

We take this opportunity to express our sincere thanks and deep gratitude to all those people who extended their wholehearted co-operation and helped us in completing this final year project successfully.

First of all, we would like to thank **Mr. Ashok Kumar** for all the help and guidance extended to us by him in every stage during the duration of our project. His inspiring suggestions and timely guidance enabled us to perceive various aspects of the project in a new light. Working under him was an enriching experience. We express our sincere thanks to **Prof. (Dr.) Prashant Johari** for her encouragement and valuable suggestions.

In all we found a congenial work environment at Galgotias University and this completion of the project will mark a new beginning for us in the coming days.

Signature of student

(Abhishek Verma)

LIST OF FIGURE

Figure Title

LIST OF TABLE

TableTitle

ABSTRACT

KEYWORDS: *Cryptography, Data Confidentiality, Comparison, Multi-cloud, Security*

Cloud Computing is a new paradigm of distributed computing which provides data and other resources to users on demand over the network. Cloud applications offer reliability, utility, scalability and availability. A lot of cloud servers provide cloud services. The cloud provides a number of services like Software as a service (SaaS), Platform as a service (PaaS), and Infrastructure as a service (IaaS). Each service has their own associated security issue. With this comes a number of challenges in its implementation. In today's world of internet technology that covers especially communication network security is a challenging issue. Hackers try to gain control over our system and steel data from it. To avoid this providing network security is an important task. Cryptography along with its various methods is used to serve this purpose. Cryptography is a technique to protect message by transforming it into an unreadable format called cipher text. It provides authentication, identification to user data, confidentiality and also provides security and privacy to the data stored. This paper focuses on the major challenge on Security & Privacy related with Cloud Computing.

CHAPTER 1

INTRODUCTION

Cloud computing is a recently developing paradigm of distributed computing. Cloud computing is a computing paradigm, where a large pool of systems are connected in private or public networks, to provide dynamically scalable infrastructure for application, data and file storage. With the advent of this technology, the cost of computation, application hosting, content storage and delivery is reduced significantly.

Cloud computing is one of the fastest growing technologies in the field of computation and storage of data and files. Cloud computing refers to manipulating, configuring and accessing the resources remotely. It offers online data storage, infrastructure and application on demand of the client with minimal management effort. It is a kind of Internet-based computing that provides shared processing resources and data to computers and other devices on demand.

There are three basic services models [1] –

- infrastructure as a service (IaaS),
- platform as a service (PaaS) and
- Software as a service (SaaS), working behind the scene making the cloud computing feasible and accessible to the users.

There are several attributes of cloud computing which allows users to deploy the cloud services. These are as follows:

1) ***Reliability***: Cloud computing is a reliable and consistent option when there is a managed service platform.

2) ***Scalability***: It provides the ability to scale the bandwidth and storage space of the system.

3) ***Elasticity***: Users can rapidly increase and decrease their computing resources as needed.

4) ***On-demand service***: Users are provided the services according to their demand and are allowed to pay for only the resources they actually use for a particular time frame.

Along with all the advantages of the cloud services, the main area of concern in the field of cloud computing is security. Cloud Computing stores the data in the open

environment at external servers which makes the data prone to potential attacks. The data owners feel unsafe to store their data on cloud. The cloud provider should ensure that the architecture and the infrastructure used by them is secure and the data and applications of the client are not compromised. Even at the time of transmission of data it is required that the data is not prone to any kind of active or passive attack.

One way of resolving this issue is encryption. The main concern of encryption is to provide-

1) Confidentiality

2) Data integrity

3) Authentication

4) Non Repudiation

There are various algorithms [2] used for the encryption. These are classified into two categories-

1) ***Symmetric Key Algorithms***: Symmetric key algorithms use single shared key to encrypt as well to decrypt the data. The sender encrypt the data using a key and then send the cipher text to the receiver. The receiver uses the same shared key to decrypt the data.

2) ***Asymmetric Key Algorithms****:* Asymmetric key algorithms use two different keys for encryption and decryption processes. These keys are known as public key and private key. The sender encrypt the data using the public key of receiver and send the cipher text to the receiver. The receiver then decrypt the cipher text using its private key. As compared to symmetric key algorithms, asymmetric key algorithms are more secured.

The encryption can be done at server side as well as at client side.

1) ***Server side encryption*****:** When data is sent to the service provider then the data is encrypted first and then stored at cloud. But this scheme of encryption is more vulnerable as data can be attacked in the way to the service provider which will result in loss of integrity of data.

2) ***Client side encryption****:* Another way of maintaining data integrity is that the data should be encrypted at client side before sending it to the service provider

There are several techniques of encryption which are proposed to ensure the security requirement of data but they are still prone to security breach. To address this issue we have proposed a new scheme of encryption of data at client side which is highly secure and is meant for storage of highly confidential data.

In 1969 [16] L. Kleinrock anticipated,
As of now, computer networks are still in their infancy. But as they grow up and become more sophisticated, we will probably see the spread of 'computer utilities' which, like present electric and telephone utilities, will service individual homes and offices across the country." His vision was the true indication of today's utility based computing paradigm. One of the giant steps towards this world was taken in mid 1990s when grid computing was first coined to allow consumers to obtain computing power on demand. The origin of cloud computing can be seen as an evolution of grid computing technologies. The term Cloud computing was given prominence by Google's CEO Eric Schmidt in late 2006 (may be he coined the term) [6]. So the birth of cloud computing is very recent phenomena although its root belongs to some old ideas with new business, technical and social perspectives. From the architectural point of view cloud is naturally build on an existing grid based architecture and uses the grid services and adds some technologies like virtualization and some business models.

In brief cloud is essentially a bunch of commodity computers networked together in same or divergent geographical locations, operating together to serve a number of customers with divergent need and workload on demand basis with the help of virtualization. Cloud services are provided to the cloud users as utility services like water, electricity, telephone using pay-as-you-use business model. These utility services are generally described as XaaS (X as a Service) where X can be Software or Platform or Infrastructure etc. Cloud users use these services provided by the cloud providers and build their applications in the internet and thus deliver them to their end users. So the cloud users don't have to worry about installing, maintaining hardware and software needed. And they also can afford these services as they have to pay as much they use. So the cloud users can reduce their expenditure and effort in the field of IT using cloud services instead of establishing IT infrastructure themselves. Cloud is essentially provided by large distributed data centres. These data centres are often organized as grid and the cloud is built on top of the grid services.

Cloud users are provided with virtual images of the physical machines in the data centres. This virtualization is one of the key concept of cloud computing as it essentially builds the abstraction over the physical system. Many cloud applications are gaining popularity day by day for their availability, reliability, scalability and utility model. These applications made distributed computing easy as the critical Cloud Computing aspects are handled by the cloud provider itself. Cloud computing is growing now-a-days in the interest of technical and business organizations but this can also be beneficial for solving social issues. In the recent time E-Governance is being implemented in developing countries to improve efficiency and effectiveness of governance. This approach can be improved much by using cloud computing instead of traditional ICT. In India, economy is agriculture based and most of the citizens live in rural areas. The standard of living, agricultural productivity etc. can be enhanced by utilizing cloud computing in a proper way. Both of these applications of cloud computing have technological as well as social challenges to overcome.

In this report we would try to clarify some of the ideas {Why is cloud computing a buzzword today? i.e. what are the benefits the provider and the users get using cloud? Though its idea has come long back in 1990 but what situation made it indispensable today? How is cloud built? New approach to implement in the cloud? What are the different services provided by the cloud providers? Though cloud computing now-a-days talks about business enterprises not the non-profit organizations; how can this new paradigm be used in the services like e-governance and in social development issues of rural India?

Once a cloud is established, how its cloud computing services are deployed in terms of business models can differ depending on requirements. The primary service models being deployed are commonly known as:

Software as a Service (SaaS)

In this model, a complete application is offered to the customer, as a service on demand. A single instance of the service runs on the cloud & multiple end users are serviced. On the customers‟ side, there is no need for upfront investment in servers or software licenses, while for the provider, the costs are lowered, since only a single application needs to be hosted & maintained. Today SaaS is offered by companies such as Google, Salesforce, Microsoft, Zoho, etc.

Platform as a Service (PaaS)

Here, a layer of software, or development environment is encapsulated & offered as a service, upon which other higher levels of service can be built. The customer has the freedom to build his own applications, which run on the provider‟s infrastructure. To meet manageability and scalability requirements of the applications, PaaS providers offer a predefined combination of OS and application servers, such as LAMP platform (Linux, Apache, MySql and PHP), restricted J2EE, Ruby etc. Google‟s App Engine, Force.com, etc are some of the popular PaaS examples.

Infrastructure as a Service (IaaS)

IaaS provides basic storage and computing capabilities as standardized services over the network. Servers, storage systems, networking equipment, data center space etc. are pooled and made available to handle workloads. The customer would typically deploy his own software on the infrastructure. Some common examples are Amazon, GoGrid, 3 Tera, etc.

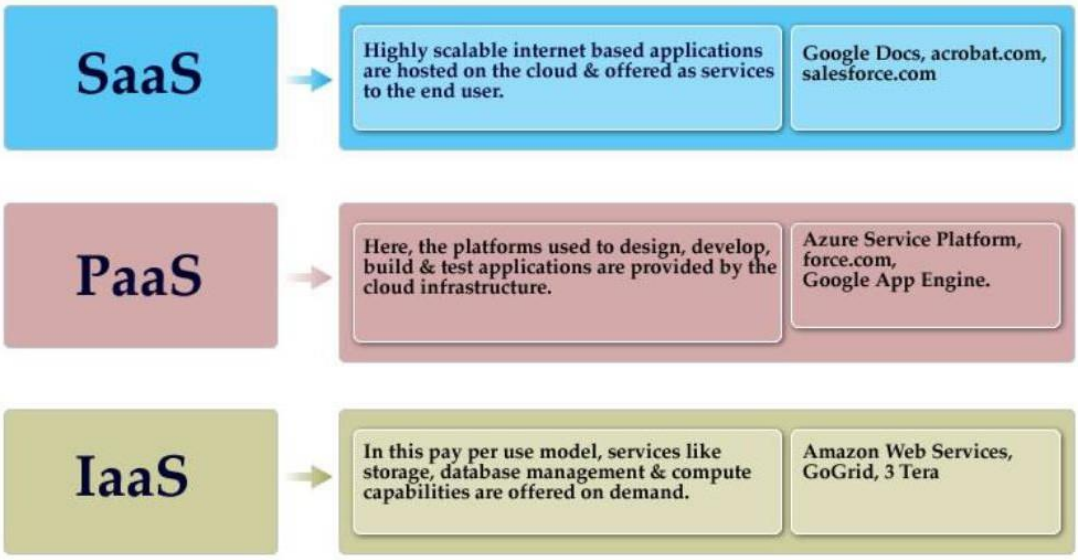


Fig.1.1 Cloud Models

Benefits of using cloud:

Enterprises would need to align their applications, so as to exploit the architecture models that Cloud Computing offers. Some of the typical benefits are listed below:

1. Reduced Cost

There are a number of reasons to attribute Cloud technology with lower costs. The billing model is pay as per usage; the infrastructure is not purchased thus lowering

maintenance. Initial expense and recurring expenses are much lower than traditional computing.

2. Increased Storage

With the massive Infrastructure that is offered by Cloud providers today, storage & maintenance of large volumes of data is a reality. Sudden workload spikes are also managed effectively & efficiently, since the cloud can scale dynamically.

3. Flexibility

This is an extremely important characteristic. With enterprises having to adapt, even more rapidly, to changing business conditions, speed to deliver is critical. Cloud computing stresses on getting applications to market very quickly, by using the most appropriate building blocks necessary for deployment.

CHAPTER 2

LITERATURE SURVEY

INTRODUCTION

Data owner depends upon Cloud Computing for various computing applications and data storage. Here, the security of data is wholly in the hands of Cloud Service Provider. The cloud service provider should ensure that proper mechanism are applied for the security of the cloud storage. Cloud service providers ensure the security at server side and try to claim that they have taken efficient and appropriate measures to secure their cloud storage. But there have been many attacks on various cloud service providers so the data security at client side becomes equally important. When data owners outsource the data to cloud, data confidentiality and data integrity are the two main factors which should we kept in mind. So to ensure security at client side various attempts are made to design various schemes. In [3], Tania Gaur and Divya Sharma proposed a scheme of applying cryptographic technique of encrypting the data at client side before outsourcing it to the cloud. They make use of the concept of the Diffie-Hellman algorithm to generate a shared key which is used for encryption and decryption. The used AES (Advanced Encryption Standard) algorithm to encrypt the data using the key generated by the Diffie-Hellman algorithm. This data is then uploaded to the cloud and the key is shared at network through Third Party Auditor (TPA). In[4], a Virtual Machine is provided through the means of Software as a Service which encrypt the data using RSA algorithm and SHA is used to create message digest of this encrypted data is sent to Cloud using Third Party Auditor (TPA). The scheme proposed in [6][7][8], also shows the use of third party auditor to store the data to the cloud. But when we deal with highly confidential data, we cannot rely on Third Party Auditor (TPA). These schemes are vulnerable when Third Party Auditor (TPA) is not trustworthy. In [5], both symmetric and asymmetric encryption techniques are used. The data is encrypted using Blowfish algorithm and then the encrypted data is converted to message digest using MD5 technique. In [7], the proposed scheme uses the Byte Rotation Encryption Algorithm (BREA) which decomposes the data into blocks of 16 bytes and then monoalphbetic substitution concept in followed and byte rotation scheme is used [10] which describes various

encryption technique such as Advanced Encryption Standard (AES), Hybrid Vigenere Caesar Cipher Encryption (HVCCE), Fully Homomorphic Encryption, Hierarchical Identity Based Encryption (Hibe). In [12], two level of security is applied using DES and RSA. At first level of encryption DES is used to encrypt data and then this encrypted data is again encrypted using RSA algorithm. Comparative study of [12][13][14] shows the use of various symmetric and asymmetric algorithm used for encryption the data. These proposed schemes either use single encryption algorithm or uses single cloud to store the data. In the worst case, if the attacker is able to guess the algorithm used for encryption by repeatedly analysing the pattern of the cipher text then the confidential data will get exposed.

The performance evaluation of different symmetric and asymmetric encryption algorithms was extensively study in the literature. Seth et. al. [1] conducted a comparative analysis for the performance evaluation of symmetric and asymmetric encryption algorithms i.e. AES, DES and RSA in term of computation time, memory usage and output bytes on different file sizes. The result of their experiments showed that DES algorithm performed better among others in term of encryption time, AES has least memory usage and RSA algorithm generated least output file. Challa et. al. compared the performance of RSA and NTRU asymmetric algorithms on variable text file sizes with the key size of 51 bits and 20 bits for encryption and decryption process respectively [5]. They concluded that NTRU performed better in term of encryption, decryption and authentication than RSA. Vijayalakshmi et. al. compared the performance of RSA and Elliptic Curve Cryptosystem (ECC) asymmetric algorithms over execution time and memory size for encryption and decryption process with variable word lengths and different key sizes. Their results showed the superiority of ECC over RSA in term of execution time and memory requirement [9]. Elminaam et. al. conducted various experiment for performance evaluation of symmetric algorithms i.e. AES (Rijndael), DES, 3DES, RC2, Blowfish, and RC6 encryption in term of energy, power consumption, different key sizes, data types and packet size in [7]. Their results showed that Blowfish performed better than other encryption algorithms on variable packet size. Elminaam et. al. compared the performance of these algorithms on different sizes of data blocks, different data types, battery power consumption, data transmission over wireless network [8] and experimental results showed that Blowfish again performed better on variable packet size. Mittal conducted

a study to compare the performance in term of processing time and throughput of symmetric algorithms i.e. DES, 3DES, and AES (Rijndael) algorithms over different in [6]. AES (Rijndael) algorithm has shown lesser execution time as compared to other algorithms and Processor 2.00 GHz (dual) showed best throughput over other hardware processors

MODEL AND ASSUMPTIONS

We have assumed that the model is composed of three entities namely, Data owner (DO), Cloud service provider (CSP) and the User. We are assuming that the cloud is highly secure at its architecture. In order to exchange data and store it on cloud, the user needs to get itself registered at DO. As a result DO shares encryption keys, algorithms, index-hash mapping table and other relevant information. Only upon successful authentication can the data be stored and retrieved from the cloud.

In our paper 1:

We have analyzed DES, AES RIJNDAEL, AES SERPENT, and BLOWFISH closely to find the best algorithm to be used. Based on our study we concluded that AES RIJNDAEL is the best encryption algorithm that can be implemented on cloud. We have made an application in java using NETBEANS IDE that implements this algorithm on files for storage on cloud and its subsequent retrieval from the cloud. We have also made an app for the same so that the same can be accessed from web as well as mobile. We have used ECLIPSE IDE for making our android app. To depict the implementation of cloud we are using WAMP server. We have saved our SQL query in it. For future work, to increase security we can break the file in to parts and implement AES RIJNDAEL on first part and AES SERPENT on second part of the file. This will enhance the level of security in our application. We have one more approach to increase the security of file. We can apply two different encryption algorithms on same file based on its size. If the size of file is even we can apply AES RIJNDAEL else AES SERPENT. The outsider would never be able to know what algorithm might have been applied, since he is unaware of the size of the file uploaded on cloud.

In our paper 2:

To ensure a secure communication we have used the concept of application of two encryption algorithms based on their even odd index position. We have also proposed making an index-hash mapping table to ensure that the segments can be rearranged at

the time of decryption. The data segments are encrypted with the hash values before sending it to the cloud. This increases the security of the encrypted data segments. Also the strategy to employ encryption is changed here. We are applying encryption on individual segments separately. This feature brings in enhanced security. The time happens to be a slight concern. Since the data is assumed to be highly confidential and important, a time trade off can be accepted at the expense of high level of security obtained. Even after capturing a segment or two, the hacker will not be able to identify the encryption technique used in them. It becomes very difficult to decrypt the entire message. The data is stored securely on multi-cloud. Here by multi cloud, we mean that the single cloud service provider has many sub clouds at scattered places. The data on multi-cloud is changed dynamically at regular intervals to ensure that the security of data is not breached at any cost.

We have presented a complete model for secure communication between different entities and secure access to data. There are four algorithms in the proposed scheme. Algorithm 1 describes the procedure to break the file into segments. Algorithm 2 describes the technique to calculate the hash values of individual segments using MD5[15] hashing technique. Algorithm 3 describes the application of two different encryption algorithms i.e. Rijndael[16] and Serpent[16] on segments by finding odd even index values. Algorithm 4 describes secure communication of data between DO, CSP and the storage at cloud. Lastly, the reverse of above procedure occurs after checking the user's authorization to retrieve a file.

Our algorithm accepts a data of any size and then breaks the data file into segments. We have used a ceiling function of 2n to break the data. Once the data is broken into segments, an index-hash mapping table is created. Hash value corresponding to every index value is calculated using MD5 hashing technique. This is then stored in the table by Data owner and shared with the user. The odd index valued data segments are encrypted using Rijndael algorithm and the even index valued data segments are encrypted using Serpent algorithm. We have used the two best symmetric key algorithms to ensure high level of security. If the hacker happens to capture a data segment then he might not be able to decrypt the entire message as it is only a part of it. Also it would be difficult for him to decrypt as he might not know which of the two algorithms were used to encrypt it.

The segments after being encrypted individually are sent to the cloud for secure storage. Concept of multi- cloud storage has been used to store the data on cloud. Even on the multi cloud, we have suggested dynamic position changes of the segments (inter and intra cloud).

When the user wishes to fetch the data from the cloud. He will be required to send the hash values of the segments to the cloud. Cloud service provider sends the data segments corresponding to the demanded hash values to the user. Upon receiving the segments the user rearranges the segments using the index-hash values table. Then the appropriate decryption algorithm is applied.

OVERVIEW OF TYPE OF CLOUD MODELS

Cloud can be of three types-

Private Cloud

This type of cloud is maintained within an organization and used solely for their internal purpose. So the utility model is not a big term in this scenario. Many companies are moving towards this setting and experts consider this is the 1st step for an organization to move into cloud. Security, network bandwidth are not critical issues for private cloud.

Private clouds are built exclusively for a single enterprise. They aim to address concerns on data security and offer greater control, which is typically lacking in a public cloud. There are two variations to a private cloud:

- **On-premise Private Cloud:** On-premise private clouds, also known as internal clouds are hosted within one's own data center. This model provides a more standardized process and protection, but is limited in aspects of size and scalability. IT departments would also need to incur the capital and operational costs for the physical resources. This is best suited for applications which require complete control and configurability of the infrastructure and security.

- **Externally hosted Private Cloud:** This type of private cloud is hosted externally with a cloud provider, where the provider facilitates an exclusive cloud environment with

full guarantee of privacy. This is best suited for enterprises that don't prefer a public cloud due to sharing of physical resources.

Public Cloud

In this type an organization rents cloud services from cloud provider's on-demand basis. Services provided to the users using utility computing model. Public clouds are owned and operated by third parties; they deliver superior economies of scale to customers, as the infrastructure costs are spread among a mix of users, giving each individual client an attractive low-cost, "Pay-as-you-go" model. All customers share the same infrastructure pool with limited configuration, security protections, and availability variances. These are managed and supported by the cloud provider. One of the advantages of a Public cloud is that they may be larger than an enterprises cloud, thus providing the ability to scale seamlessly, on demand.

Hybrid Cloud

This type of cloud is composed of multiple internal or external cloud. This is the scenario when an organization moves to public cloud computing domain from its internal private cloud.Enterprises can choose to deploy applications on Public, Private or Hybrid clouds. Cloud Integrators can play a vital part in determining the right cloud path for each organization.

Hybrid Clouds combine both public and private cloud models. With a Hybrid Cloud, service providers can utilize 3rd party Cloud Providers in a full or partial manner thus increasing the flexibility of computing. The Hybrid cloud environment is capable of providing on-demand, externally provisioned scale. The ability to augment a private cloud with the resources of a public cloud can be used to manage any unexpected surges in workload.

CLOUD STAKEHOLDERS

To know why cloud computing is used let's first concentrate on who use it. And then we would discuss what advantages they get using cloud. There are three types of stakeholders cloud providers, cloud users and the end users. Cloud providers provide cloud services to the cloud users. These cloud services are of the form of utility

computing i.e. the cloud users uses these services pay-as-you-go model. The cloud users develop their product using these services and deliver the product to the end users.

ADVANTAGES OF USING CLOUD

The advantages for using cloud services can be of technical, architectural, business etc. [5, 6].

Cloud Providers' point of view

(a) Most of the data centers today are underutilized. They are mostly 15% utilized. These data centers need spare capacity just to cope with the huge spikes that sometimes get in the server usage. Large companies having those data centers can easily rent those computing power to other organizations and get profit out of it and also make the resources needed for running data center (like power) utilized properly.

(b) Companies having large data centers have already deployed the resources and to provide cloud services they would need very little investment and the cost would be incremental.

Cloud Users' point of view

(a) Cloud users need not to take care about the hardware and software they use and also they don't have to be worried about maintenance. The users are no longer tied to someone traditional system.

(b) Virtualization technology gives the illusion to the users that they are having all the resources available.

(c) Cloud users can use the resources on demand basis and pay as much as they use. So the users can plan well for reducing their usage to minimize their expenditure.

(d) Scalability is one of the major advantages to cloud users. Scalability is provided dynamically to the users. Users get as much resources as they need. Thus this model perfectly fits in the management of rare spikes in the demand.

CHALLENGES IN CLOUD COMPUTING

Despite its growing influence, concerns regarding cloud computing still remain. In our opinion, the benefits outweigh the drawbacks and the model is worth exploring. Some

common challenges are:

1. Data Protection

Data Security is a crucial element that warrants scrutiny. Enterprises are reluctant to buy an assurance of business data security from vendors. They fear losing data to competition and the data confidentiality of consumers. In many instances, the actual storage location is not disclosed, adding onto the security concerns of enterprises. In the existing models, firewalls across data centers (owned by enterprises) protect this sensitive information. In the cloud model, Service providers are responsible for maintaining data security and enterprises would have to rely on them.

All business applications have Service level agreements that are stringently followed. Operational teams play a key role in management of service level agreements and runtime governance of applications. In production environments, operational teams support

- Appropriate clustering and Fail over
- Data Replication System monitoring (Transactions monitoring, logs monitoring and others)
- Maintenance (Runtime Governance)
- Disaster recovery
- Capacity and performance management

If, any of the above mentioned services is under-served by a cloud provider, the damage & impact could be severe.

2. Management Capabilities

Despite there being multiple cloud providers, the management of platform and infrastructure is still in its infancy. Features like „Auto-scaling" for example, are a crucial requirement for many enterprises. There is huge potential to improve on the scalability and load balancing features provided today.

3. Regulatory and Compliance Restrictions

In some of the European countries, Government regulations do not allow customer's personal information and other sensitive information to be physically located outside the state or country. In order to meet such requirements, cloud providers need to setup a data center or a storage site exclusively within the country to comply with regulations.

Having such an infrastructure may not always be feasible and is a big challenge for cloud providers.

With cloud computing, the action moves to the interface — that is, to the interface between service suppliers and multiple groups of service consumers. Cloud services will demand expertise in distributed services, procurement, risk assessment and service negotiation — areas that many enterprises are only modestly equipped to handle.

PERFORMANCE EVALUATION

Table 2.1 Comparison of AES and DES

Factors	AES	DES
Developed	2000	1977
Key Size	128, 192, 256 bits	56 bits
Block Size	128 bits	64 bits
Ciphering & deciphering key	Same	Same
Scalability	Not Scalable	It is scalable algorithm due to varying the key size and Block size.
Algorithm	Symmetric Algorithm	Symmetric Algorithm
Encryption	Faster	Moderate
Decryption	Faster	Moderate
Power Consumption	Low	Low
Security	Excellent Secured	Not Secure Enough
Deposit of keys	Needed	Needed
Inherent Vulnerabilities	Brute Forced Attack	Brute Forced, Linear and differential cryptanalysis attack
Key Used	Same key used for Encrypt and Decrypt	Same key used for Encrypt and Decrypt
Rounds	10/12/14	16
Stimulation Speed	Faster	Faster
Trojan Horse	Not proved	No
Hardware & Software Implementation	Faster	Better in hardware than in software
Ciphering & Deciphering Algorithm	Different	Different

Based on the comparisons made on different cryptographic algorithms such as AES, DES, Blowfish, and SERPENT etc., we found that in this internet world, security of data play a major role as data and communications are passed and are done over open networks. From our evaluation, we found that in cryptographic algorithms symmetric encryption technique and asymmetric encryption techniques both have a higher ratio for encryption. [1][2]The key size is higher in asymmetric algorithm because of which in case of changing key size, it is viewed that higher key size leads to clear change in

the battery and time consumption. [4][5] In symmetric techniques DES is considered to be the most secured because the most efficient attack on it is still now brute force attack and AES is proven to be fast in both hardware and software. But if we see the overall performance of all symmetric algorithms AES is viewed as a better solution. Three fish a derived algorithm from blowfish consumes less power and it is also fast.[6] Each algorithm is unique in its own way and they are useful in real time encryptions. Each one is suitable in different applications, so it depends on the user to select the most appropriate algorithm that is best suited to his needs. Through our analysis on different cryptographic algorithms we wish to provide a pathway for future researchers to promote the performance and security of cryptographic algorithms.

ANDROID ENCRYPTION SYSTEMS

Encryption is the process of encoding user data on an Android device using an encrypted key. Once a device is encrypted, all user-created data is automatically encrypted before committing it to disk and all reads automatically decrypt data before returning it to the calling process. Android is a modern mobile platform that was designed to be truly open. Android applications make use of advanced hardware and software, as well as local and served data, exposed through the platform to bring innovation and value to consumers. To protect that value, the platform must offer an application environment that ensures the security of users, data, applications, the device, and the network.

Securing an open platform requires a robust security architecture and rigorous security programs. Android was designed with multi-layered security that provides the flexibility required for an open platform, while providing protection for all users of the platform.

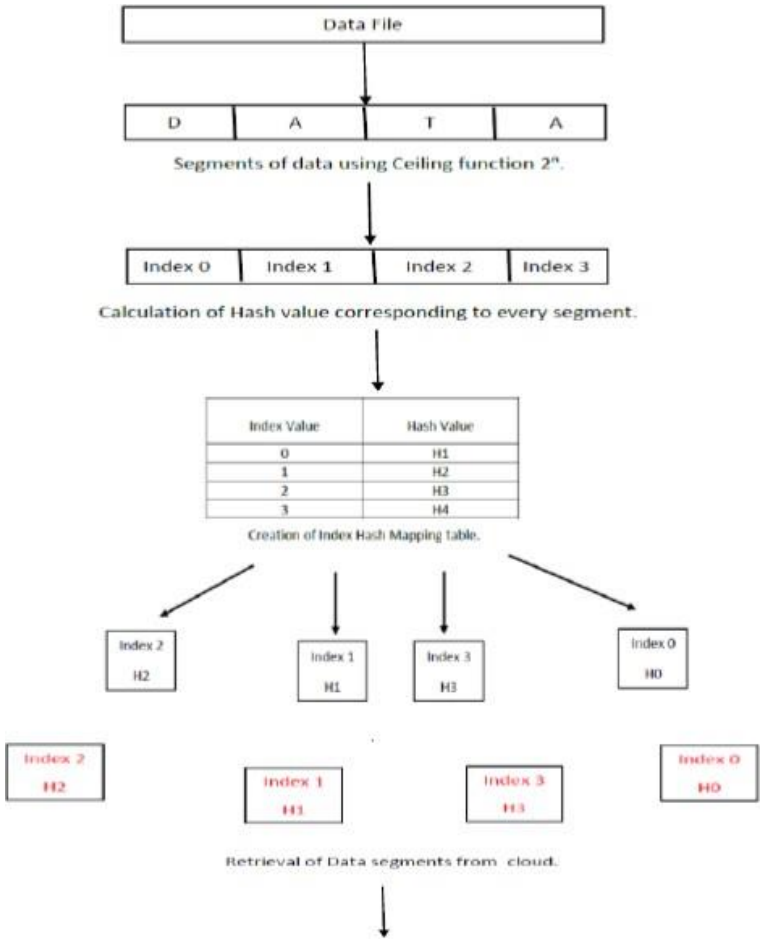
Android was designed with developers in mind. Security controls were designed to reduce the burden on developers. Security-savvy developers can easily work with and rely on flexible security controls. Developers less familiar with security will be protected by safe defaults.

Android was designed with device users in mind. Users are provided visibility into how applications work, and control over those applications. This design includes the expectation that attackers would attempt to perform common attacks, such as social

engineering attacks to convince device users to install malware, and attacks on third-party applications on Android. Android was designed to both reduce the probability of these attacks and greatly limit the impact of the attack in the event it was successful.

Cryptography technique needs some algorithm for encryption of data. The encryption/decryption process refers to the operation of dividing and replacing an arrangement of the original image .the image can be decomposed into blocks; each one contains a specific number of pixels. The blocks are transferred into new locations. For better process the block size should be small, because fewer pixels keep the neighbours. In this case the correlation will be decreased and thus it becomes difficult to predict the value of any given pixel from the values of its neighbours. At the receiver side, the original image can be obtained by the inverse transformation of the blocks.

Based on the assessment criteria defined above, Android's encryption systems are discussed and assessed in this section to derive potential attack scenarios.



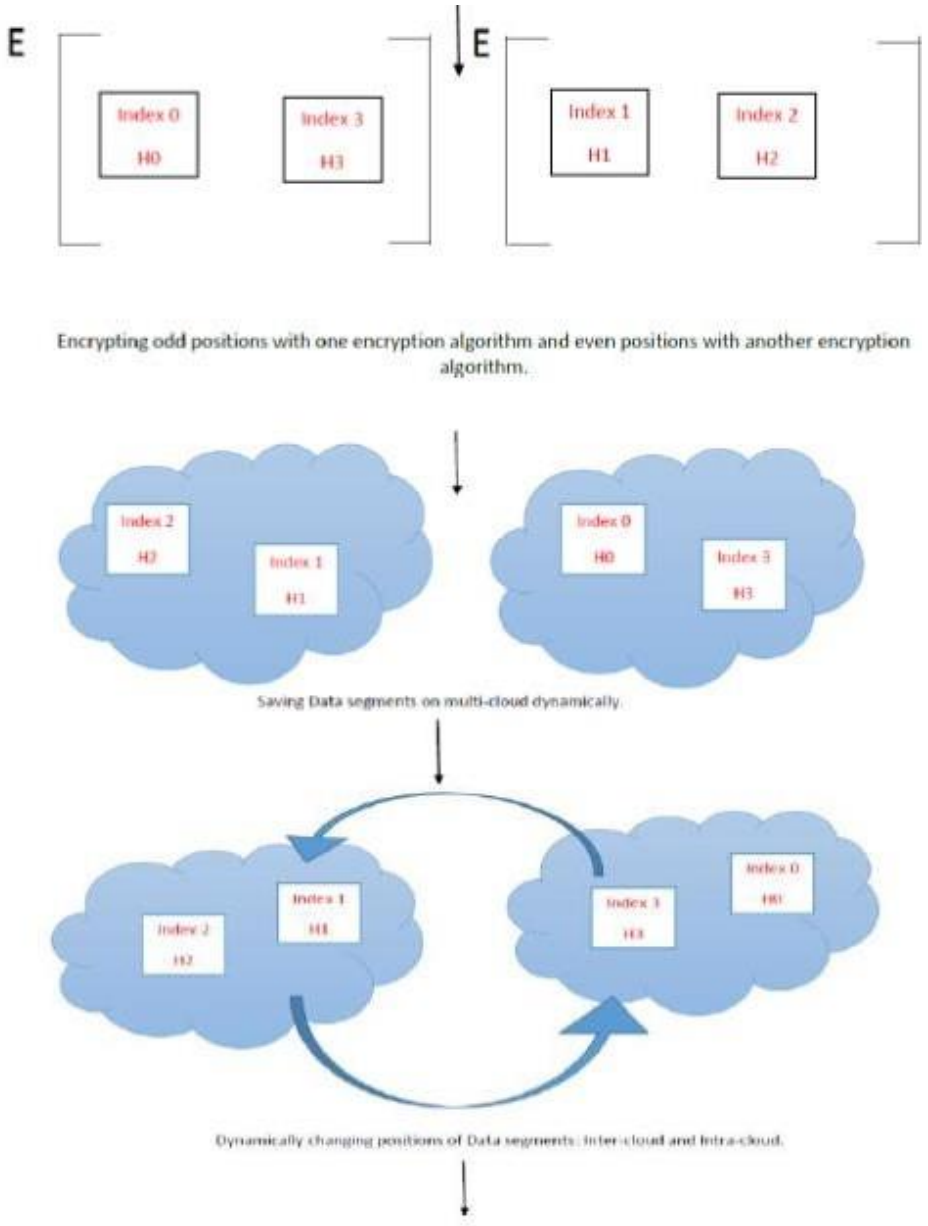


Fig. 2.1 Overview of the flow of proposed scheme

CHAPTER 3

PROPOSED WORK

Based on the studies and research conducted, we discovered that:

There is large need of an application that protects user's data from geographically any coordinate and allows them to conveniently store and retrieve data from the cloud.

The users face problem to manage the security and integrity of their data because of the security issues wherein the data become extremely vulnerable to be hacked from any part of the world by any person and in order to avoid it we needed mechanisms that protect the data comprehensively. There is a large need of an application that implements a large scale of encoding mechanisms which also involve obscurity techniques because encoding and hiding of data are two very strong and significant aspects to protect data.

There is a large need of a simple system that does not involve much user interaction and is able to perform the tasks on its own. Users nowadays want convenience that their data should protected wherever it is, be it handset, network, cloud, data centre or anywhere and our application shall ensure that it is done and the users do not have to concern themselves for manual intervention regarding protection of their sensitive data. They want that the data should be protected automatically, conveniently and efficiently so they do not think about decoding or hacking by any unauthorised authority.

There is a large need of a system that simplifies work, that enables the user to access a simple application while on-the-go and facilitates him/her to save as well securely retrieve data using simplified yet very robust encoding and obscurity mechanisms.

In our proposed implementation:

- There would be 2 tiers of the project namely Android and Cloud.
- Users (Sender & Receiver) shall register themselves on http://www.parse.com cloud service for obtaining user id
- Sender will login the Android application with user id and password of the account created on cloud.
- After login, sender shall upload the file.

- The uploaded file will be encrypted by the Android application using **AES** algorithm.
- The encrypted image will further be secured by an encrypted password key, set by the sender through the Android application.
- The doubly-secured, encrypted file is ready to be sent to the cloud.
- Sender transmits the file to the cloud.
- When the sender has completed the transmission, it is the responsibility of cloud to protect it.
- When the sender has sent, it gets saved on the cloud database in an encrypted form which disallows it to be viewed even on the cloud.
- In order to view the file, the recipient taps on the file in the file name list and inputs the respective password in order to decrypt the file.
- Thee file can then be viewed upon entering the right key.
- If the user enters a wrong key, he might not be able to see the correct file.

CHAPTER 4

METHODOLOGY

To better manage the development process and to achieve consistency, it is essential that the software development be done in phases.

We shall develop this system after duly spending time on each software development phase individually and freezing the status before we move on to the next phase i.e. we will use *Linear Sequential* model in this application under which:

1. In the analysis phase, we will attempt to understand the system completely in terms of its objectives & the problems faced. All the objectives will further be subdivided into a set of smaller objectives which in turn were subdivided in to the set of actions. The analysis will also give us an overview about the individual expectations from each function and the challenges faced by that function due to which the ultimate objective cannot be achieved.

2. Before we move to the designing phase, the objectives & challenges will be clearly understood and actually converted into the set of objectives & actions into modules. The modules will be designed in terms of their input, output, flow of information, storage of information & communication amongst each other, with the user and with the system. All the data objects will be carefully designed and classified in terms of their inputs & outputs.

3. After the analysis & design phases were over, we shall move on to coding phase, where the implementation of tasks or functions on-paper will be actualized. This will be the phase where we actually come to see the look & feel of the application, where we actually think from the user's perspective & company's perspective keeping all the objectives & challenges into consideration.

4. Then testing phase shall come in which after developing the complete system, we shall rigorously test it using all the testing types and checked every characteristic / attribute of the application of whether it coincides and is in line with the company's objectives and user's comfort.

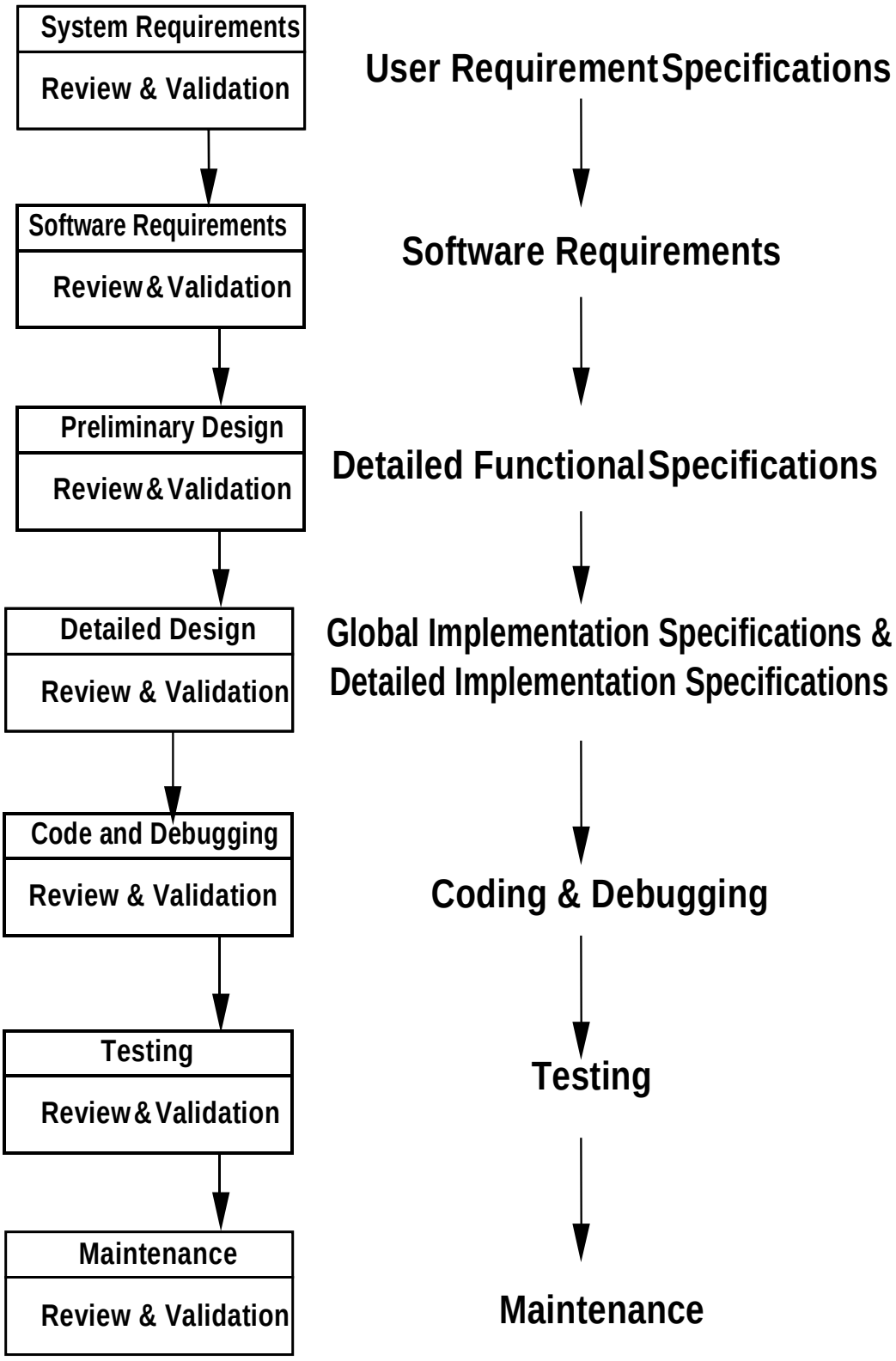


Fig. 4.1 Work Flow

CHAPTER 5

IMPLEMENTATION

Use Case Diagram:

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

In software and systems engineering, a use case is a list of action or event steps, typically defining the interactions between a role (known in the Unified Modeling Language as an *actor*) and a system, to achieve a goal. The actor can be a human, an external system, or time. In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or stake holder goals.

Use case analysis is an important and valuable requirement analysis technique that has been widely used in modern software engineering since their formal introduction by Ivar Jacobson in 1992.

Use case driven development is a key characteristic of many process models and frameworks such as ICONIX, the Unified Process (UP), the IBM Rational Unified Process (RUP), and the Oracle Unified Method (OUM). With its inherent iterative, incremental and evolutionary nature, use case also fits well for agile development.

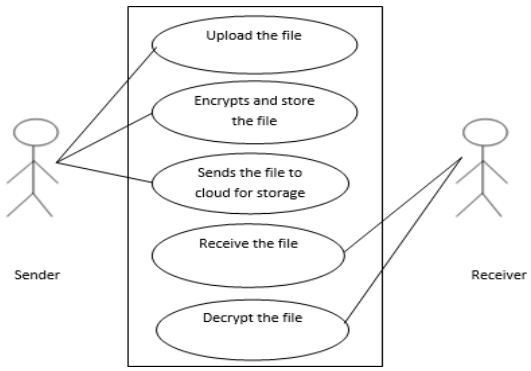


Fig. 5.1 Use Case Diagram

Flow Diagram:

Flow diagram is a collective term for a diagram representing a flow or set of dynamic relationships in a system. The term flow diagram is also used as synonym of the flowchart, and sometimes as counterpart of the flowchart.

Flow diagrams are used to structure and order a complex system, or to reveal the underlying structure of the elements and their interaction. The term flow diagram is used in theory and practice in different meanings. Most commonly the flow chart and flow diagram are used in an interchangeable way in the meaning of a representation of a process.

Flow chart or flow diagram is a diagram that visually displays interrelated information such as events, steps in a process, functions, etc., in an organized fashion, such as sequentially or chronologically. Flow diagram is a graphic representation of the physical route or flow of people, materials, paper works, vehicles, or communication associated with a process, procedure plan, or investigation.

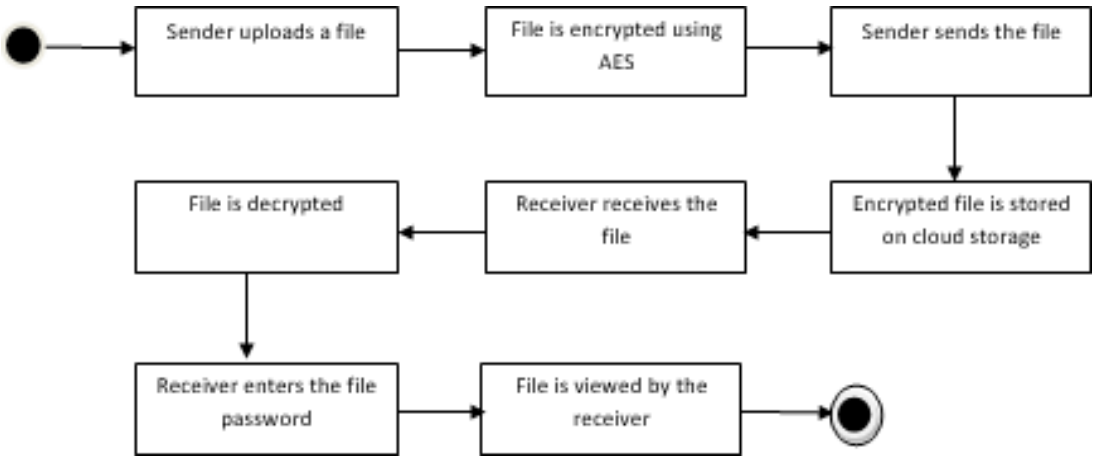


Fig. 5.2 Process Flow Diagram

Context Diagram:

A System Context Diagram (SCD) in software engineering and systems engineering is a diagram that defines the boundary between the system, or part of a system, and its environment, showing the entities that interact with it. This diagram is a high level view of a system. It is similar to a block diagram.

Context Diagrams are used early in a project to get agreement on the scope under investigation. Context diagrams are typically included in a requirements document.

These diagrams must be read by all project stakeholders and thus should be written in plain language, so the stakeholders can understand items within the document.

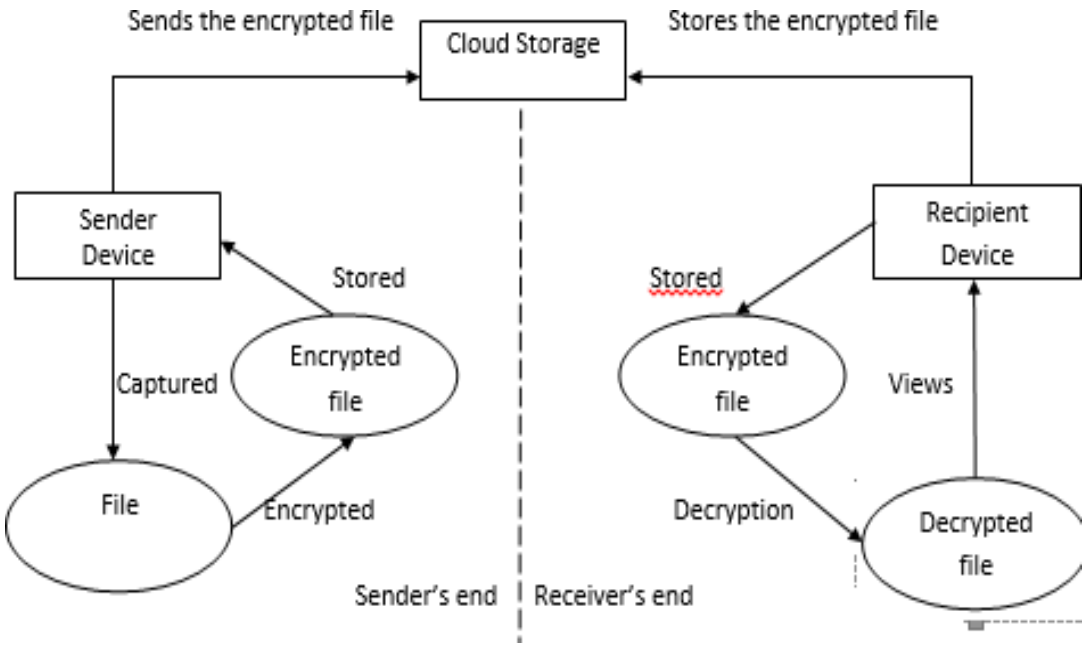


Fig. 5.3 Context Diagram

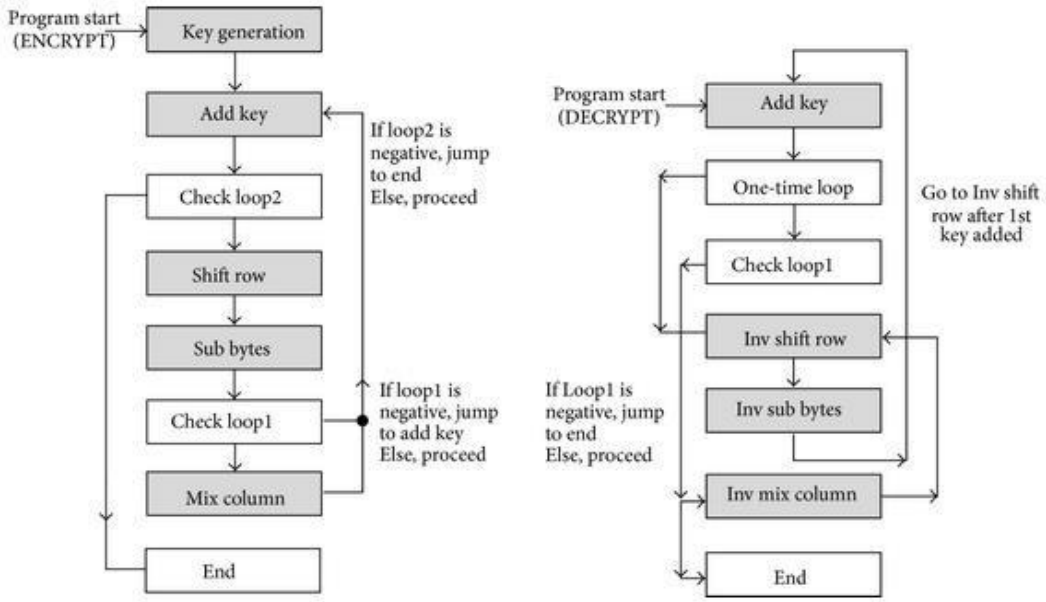


Fig. 5.4 Flowchart of AES Algorithm

In today's global competitive market, companies must innovate and get the most from its resources to succeed. Cloud computing infrastructures are next generation platforms that can provide tremendous value to companies of any size. They can help companies achieve more efficient use of their IT hardware and software investments and provide a means to accelerate the adoption of innovations. Cloud computing increases profitability by improving resource utilization. Costs are driven down by delivering appropriate resources only for the time those resources are needed. Cloud computing has enabled teams and organizations to streamline lengthy procurement processes.

Cloud computing enables innovation by alleviating the need of innovators to find resources to develop, test, and make their innovations available to the user community. Innovators are free to focus on the innovation rather than the logistics of finding and managing resources that enable the innovation.

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased. This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

An activity diagram is characterized by states that denote various operations. Transition from one state to the other is triggered by completion of the operation. The purpose of an activity is symbolized by round box, comprising the name of the operation. An operation symbol indicates the execution of that operation. This activity diagram depicts the internal state of an object.

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify

Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Field testing will be performed manually and functional tests will be written in detail.

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.
- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.Cloud computing builds on decades of research in virtualization, distributed computing, utility computing, and more recently networking, web and software service& It implies a service oriented architecture, reduced information technology overhead for the end-user, great flexibility, reduced total cost of ownership, on demand services and many other thing.

In today's global competitive market, companies must innovate and get the most from its resources to succeed. Cloud computing infrastructures are next generation platforms that can provide tremendous value to companies of any size. They can help companies achieve more efficient use of their IT hardware and software investments and provide a means to accelerate the adoption of innovations. Cloud computing increases profitability by improving resource utilization. Costs are driven down by delivering appropriate resources only for the time those resources are needed. Cloud computing has enabled teams and organizations to streamline lengthy procurement processes.
Cloud computing enables innovation by alleviating the need of innovators to find resources to develop, test, and make their innovations available to the user community. Innovators are free to focus on the innovation rather than the logistics of finding and managing resources that enable the innovation.

- Cloud enabler technologies like utility computing, Grid Computing, RTI, web infrastructure and others are cloud enabled.
- Infrastructure service providers are taking advantage of the Cloud services.

- Information services, entertainment-oriented services such as video on demand, simple business services such as customer authentication or identity management and contextual services such as location or mapping services are positioned well by using the service.
- Other services, such as corporate processes (for example, billing, deduction management and mortgage calculation) and transactional services (for example, fiscal transactions), would take longer to reach the cloud and the mainstream.
- Cloud computing infrastructures allows efficient use of their IT hardware and software investments
- A cloud infrastructure can be a cost efficient model for delivering information services, reducing IT management complexity.
- The Cloud makes it possible to launch Web 2.0 applications quickly and to scale up applications as much as needed when needed.
- Stored data might not be secure: With cloud computing, all our data is stored on the cloud. The unauthorized users gain access to our confidential data.
- Dependent on internet connection: Internet connectivity isn't completely stable and reliable.
- It's not platform agnostic: Most clouds force participants to rely on a single platform or host only one type of product.
- Can be slow: Even on a fast connection, web based application scan sometimes be slower than accessing a similar software program on our desktop PC.

In brief cloud is essentially a bunch of commodity computers networked together in same or divergent geographical locations, operating together to serve a number of customers with divergent need and workload on demand basis with the help of virtualization. Cloud services are provided to the cloud users as utility services like water, electricity, telephone using pay-as-you-use business model. These utility services are generally described as XaaS (X as a Service) where X can be Software or Platform or Infrastructure etc. Cloud users use these services provided by the cloud providers and build their applications in the internet and thus deliver them to their end users. So the cloud users don't have to worry about installing, maintaining hardware and software needed. And they also can afford these services as they have to pay as much they use. So the cloud users can reduce their expenditure and effort in the field

of IT using cloud services instead of establishing IT infrastructure themselves. Cloud is essentially provided by large distributed data centres. These data centres are often organized as grid and the cloud is built on top of the grid services.

Cloud users are provided with virtual images of the physical machines in the data centres. This virtualization is one of the key concept of cloud computing as it essentially builds the abstraction over the physical system. Many cloud applications are gaining popularity day by day for their availability, reliability, scalability and utility model. These applications made distributed computing easy as the critical Cloud Computing aspects are handled by the cloud provider itself. Cloud computing is growing now-a-days in the interest of technical and business organizations but this can also be beneficial for solving social issues. In the recent time E-Governance is being implemented in developing countries to improve efficiency and effectiveness of governance. This approach can be improved much by using cloud computing instead of traditional ICT. In India, economy is agriculture based and most of the citizens live in rural areas. The standard of living, agricultural productivity etc. can be enhanced by utilizing cloud computing in a proper way. Both of these applications of cloud computing have technological as well as social challenges to overcome.

In this report we would try to clarify some of the ideas :

Why is cloud computing a buzzword today? i.e. what are the benefits the provider and the users get using cloud? Though its idea has come long back in 1990 but what situation made it indispensable today? How is cloud built? New approach to implement in the cloud? What are the different services provided by the cloud providers? Though cloud computing now-a-days talks about business enterprises not the non-profit organizations; how can this new paradigm be used in the services like e-governance and in social development issues of rural India?

Data is the most significant entity to every being. It is the sole ingredient of human and non – human communication in all forms, be it electronic, digital, verbal or written. And because of its importance, we adopt stringent measures to secure the storage of data and ensure its authorized access at different levels so that its usability and integrity are maintained.

Security of any data deals both with its storage and retrieval. We may apply robust cryptographic algorithms in order to encode the data and/or implement several authentication checks to verify the genuineness of user attempting to access it. We wish to develop an application that enables the users to exchange classified multimedia content, securely from any geographical coordinate. We aspire to implement security at all levels, in order to shield the data, right from the point of its creation to delivery. Our intention is to be different from the existent applications under this domain, in the form of latest technology, ease-of-use and scalability.

We propose to provide convenience to the user through simple, understandable yet secure communication interfaces. Our goal is to shield date at all checkpoints through which it travels i.e. Sender Device → Network → Cloud → Recipient Device.

In this world of advanced communication, people prefer mechanisms through which data can be saved or retrieved quickly, easily and securely from any geographical coordinate. To facilitate this objective, smart phones and mobile cloud computing play an integral role. They fortify the user to use techniques for smart storage and retrieval of data using infrastructure, platform and software as a service being provided by 3rd party.

In our implementation also, we shall make use of the newest technologies, i.e. Android and Cloud Computing. Android is a software stack for mobile devices that includes an operating system, middleware and key applications. By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language.

Whereas, the term *Cloud computing* refers to the many different types of services and applications being delivered in the internet cloud, and the fact that, in many cases, the devices used to access these services and applications do not require any special applications.

Once a cloud is established, how its cloud computing services are deployed in terms of business models can differ depending on requirements. The primary service models being deployed are commonly known as:

Software as a Service (SaaS)

Software as a service (SaaS) is a software licensing and delivery model in which software is licensed on a subscription basis and is centrally hosted. It is sometimes referred to as "on-demand software". SaaS is typically accessed by users using a thin client via a web browser. SaaS has become a common delivery model for many business applications, including office and messaging software, payroll processing software, DBMS software, management software, CAD software, development software, gamification, virtualization accounting, collaboration, customer relationship management (CRM),management information systems (MIS),enterprise resource planning (ERP) ,invoicing, human resource management (HRM), talent acquisition, content management (CM), antivirus software, and service desk management. SaaS has been incorporated into the strategy of all leading enterprise software companies. One of the biggest selling points for these companies is the potential to reduce IT support costs by outsourcing hardware and software maintenance and support to the SaaS provider.

The vast majority of SaaS solutions are based on a multi-tenant architecture. With this model, a single version of the application, with a single configuration (hardware, network, operating system), is used for all customers ("tenants"). To support scalability, the application is installed on multiple machines (called horizontal scaling). In some cases, a second version of the application is set up to offer a select group of customers with access to pre-release versions of the applications for testing purposes. This is contrasted with traditional software, where multiple physical copies of the software each potentially of a different version, with a potentially different configuration, and often customized are installed across various customer sites. While an exception rather than the norm, some SaaS solutions do not use multi-tenancy, or use other mechanisms—such as virtualization—to cost-effectively manage a large number of customers in place of multi-tenancy. Whether multi-tenancy is a necessary component for software-as-a-service is a topic of controversy.

The Cloud (or SaaS) model has no physical need for indirect distribution since it is not distributed physically and is deployed almost instantaneously. The first wave of SaaS companies built their own economic model without including partner remuneration in their pricing structure (except when there were certain existing affiliations).

Platform as a Service (PaaS)

Platform as a service (PaaS) is a category of cloud computing services that provides a platform allowing customers to develop , run , and manage web applications without the complexity of building and maintaining the infrastructure typically associated with developing and launching an app. PaaS can be delivered in two ways : as a public cloud service from a provider, where the consumer controls software deployment and configuration settings , and the provider provides the networks, servers, storage and other services to host the consumer's application ; or as software installed in private data centres or public infrastructure as a service and managed by internal IT departments.

Open PaaS does not include hosting, but rather it provides open source software allowing a PaaS provider to run applications in an open source environment. For example, AppScale allows a user to deploy some applications written for Google App Engine to their own servers, providing data store access from a standard SQL or NoSQL database. Some open platforms let the developer use any programming language, database, operating system or server to deploy their applications.

The advantages to PaaS are primarily that it allows for higher-level programming with dramatically reduced complexity; the overall development of the application can be more effective, as it has built-in infrastructure; and maintenance and enhancement of the application is easier. It can also be useful in situations where multiple developers are working on a single project involving parties who are not located nearby. The primary disadvantage would be the possibility of being locked into a certain platform. However, most PaaSes are relatively lock-in-free.

Infrastructure as a Service (IaaS)

Consumers control and manage the systems in terms of the operating systems, applications, storage, and network connectivity, but do not themselves control the cloud infrastructure.

In the most basic cloud-service model - and according to the IETF (Internet Engineering Task Force) - providers of IaaS offer computers – physical or (more often) virtual machines – and other resources. IaaS refers to online services that abstract user from the detail of infrastructure like physical computing resources, location, data partitioning, scaling, security, backup etc. A hypervisor, such as Xen, Oracle Virtual Box, KVM, VMware ESX/ESXi, or Hyper-V runs the virtual machines as guests. Pools of hypervisors within the cloud operational system can support large numbers of virtual machines and the ability to scale services up and down according to customers' varying requirements. IaaS clouds often offer additional resources such as a virtual-machine disk-image library, raw block storage, file or object storage, firewalls, load balancers, IP addresses, virtual local area networks (VLANs), and software bundles. IaaS-cloud providers supply these resources on-demand from their large pools of equipment installed in data centres. For wide-area connectivity, customers can use either the Internet or carrier clouds (dedicated virtual private networks).

To deploy their applications, cloud users install operating-system images and their application software on the cloud infrastructure. In this model, the cloud user patches and maintains the operating systems and the application software. Cloud providers typically bill IaaS services on a utility computing basis: cost reflects the amount of resources allocated and consumed. The presented security analysis is based on the general scenario that a security officer of a company or public agency is in charge of deploying the Android platform to allow employees to process and store security-critical data with mobile Android devices. This is a common scenario, as mobile devices are increasingly issued to employees, in order to improve efficiency.

As mobile devices are much more likely to be subject to loss or theft than classical computing devices such as desktop PCs, theft is the main threat to be considered by the security analysis presented in this paper. Another potential threat is malware that is installed by an attacker on a mobile device to spy on security-critical data. However,

the focus is not put on this threat for several reasons. First, according to the underlying scenario, we can assume a controlled environment and hence managed mobile device that is under control of a mobile device management (MDM) solution. As for managed devices the set of installed software can be controlled by an administrator (or security officer), the risk of installed malware can be neglected. Of course, also managed devices are prone to highly sophisticated malware that exploits system vulnerabilities to gain root access to the operating system. However, this kind of malware must be assumed to have almost unlimited capabilities, including the capability to circumvent any encryption system and security feature in place. For these reasons, this work mainly focuses on the identified threat theft.

Regarding the identified threat theft, several additional assumptions apply. First, our assessments are based on the assumption that the Android encryption algorithms are implemented correctly. The goal of this assessment is to analyse weaknesses located on a higher level, such as bad configurations, weak passwords, limits of used key derivation functions, or wrong assumptions in relation to the encryption scope (e.g., files vs. file-system). Second, we assume that a passcode locked Android device is stolen by an attacker who is an expert with in-depth knowledge about the deployed encryption systems and their weaknesses. This scenario is similar to the one faced by a forensic expert who needs to analyse the data stored on an Android device. In this context we also assume that the attacker employs jail breaking/rooting tools. This is the only type of malware that will be considered in the conducted analysis.

The major concern with maintaining plausible deniability is whether the system will provide some indication of the existence of any hidden data. Mobiflage's threat model and assumptions are mostly based on past work on desktop PDE solutions (cf. TrueCrypt); we also include threats more specific to mobile devices.

Threat model and operational assumptions are-

1. Mobiflage must be merged with the default Android code stream, or a widely used custom firmware based on Android (e.g., CyanogenMod1) to ensure that many devices are capable of using PDE. Then an adversary will be unable to make assumptions about the presence of hidden volumes based on the availability of software support. We do not require a large user base to employ PDE; it is sufficient that the capability is

widespread, so the availability of PDE will not be a red flag. Similar to TrueCrypt, all installations of Mobiflage include PDE capabilities. There are no identifying technical differences between the default and PDE encryption modes. However, when more users enable default encryption, they help to obscure those that use PDE.

2. Mobiflage currently requires a physical or emulated FAT32 SD card. Devices, such as the Nexus S, which use an internal eMMC partition as opposed to a removable SD card are supported. Some devices, such as the Galaxy Nexus, have neither physical nor emulated external storage. Instead, they use the media transfer protocol (MTP) and share a single Ext4formatted partition for the (internal) app storage and (external) user accessible storage. These devices are not currently supported

3. The adversary has the encrypted device and full knowledge of Mobiflage's design, but lacks the PDE key (and the corresponding password). The offset of Mobiflage's hidden volume is dependent on the PDE password, and is therefore also unknown to the adversary.

4. The adversary has some means of coercing the user to reveal their encryption keys and passwords (e.g., unlock-screen secret), but will not continue to punish the user once it becomes futile (e.g., the adversary is convinced that he has obtained the true key, or the assurance that no such key actually exists). To successfully provide deniability in Mobiflage, the user is expected to refrain from disclosing the true key.

5. The adversary can access the user device's internal and external storage, and can have root-level access to the device after capturing it. The adversary can then manipulate disk sectors, including encryption/decryption under any decoy keys learned from the user; this can compromise deniability (e.g., the "copy-and-paste" attack). Mobiflage addresses these issues.

6. The adversary model of desktop FDE usually includes the ability to periodically access or snapshot the encrypted physical storage. However, this assumption is unlikely for mobile devices and has therefore been relaxed (as the adversary will have access to the storage media only after apprehending the user).

7. In addition to the Dolev-Yao network attacker model, we also assume that the adversary has some way of colluding with the wireless carrier or ISP (e.g., a state-run carrier, or subpoena power over the provider). Adversaries can collect network/service activity logs from these carriers to reveal the use of a PDE mode on suspected devices. This assumption significantly strengthens the attacker model, none the less, is quite realistic.

8. We assume the mobile OS, kernel, and bootloader are malware-free, and while in the PDE mode, the user does not use any adversary controlled apps to avoid leaking information via those apps; i.e., in the PDE mode, the user is expected to use only trusted apps. The device firm ware and base band OS are also trusted. Control over the baseband OS may allow an adversary to monitor calls and intercept network traffic, which may be used to reveal the PDE mode. Mobile malware, and defining/verifying trusted code are independent problems, and are out of scope here.

9. We assume the adversary cannot capture the user device while in the PDE mode; otherwise, user data can be trivially retrieved if the device is unlocked. We require the user to follow certain guidelines, e.g., not using Mobiflage's PDE-mode for regular use; other precautions are discussed later. Following these guidelines may require non-trivial effort, but is required for maintaining deniability in our threat model.

Limitations of our current Mobiflage design and prototype include the following:

1. Mobiflage currently requires a separate physical FAT32 storage partition (SD or eMMC). Devices that use MTP and share a single partition for internal and external storage are not currently supported. We discuss the problems inherent to Ext4, and provide suggestions for other file systems (e.g., HFS+, Ext2/3).

2. Users currently cannot set the desired size of a hidden volume; the size is derived from a user's password to avoid the need to store the offset on the device. An expected size may be satisfied as follows (not currently implemented). We can ask users for the desired size and iterate the hash function until an offset close to the requested size is found. For example, we can perform 20 additional hash iterations and report the closest size available with the supplied password. The user could then choose to either accept the approximate size or enter a new password and try again. Storing the iteration count

is not needed. At boot time, the system will perform consecutive iterations until a valid file system is found, or a maximum count is reached. This would slow down the boot process somewhat while searching for the correct offset.

3. Currently, we support only one hidden volume offset. Creating additional (decoy) hidden volumes will require a collision prevention mechanism to derive offsets. A method, such as the iteration count mentioned above, can be used to ensure enough space is left between hidden offsets (e.g., 1.5GB). This increases the chance of corrupting hidden data. Each hidden volume would appear to consume the remaining SD card storage, but the address space would overlap with other hidden volumes.

4. Transferring data between outer and hidden volumes may be necessary on occasion; e.g., if time does not permit switching between modes before taking an opportunistic photo. We do not offer any safe mechanism for such transfers at present. Mounting both volumes simultaneously is a straight forward solution, but may compromise deniability (e.g., usage log data of a hidden file may be visible on the decoy volume). The user can transfer sensitive files to a PC as an intermediary, then transfer the files to the PDE storage. In this case, data remanence in the outer volume is an issue. Another possibility is to keep a RAM disk mounted in the standard mode for storing such opportunistic files (and then copy to the PDE storage via a PC). However, some apps, such as the camera app, do not offer an option to choose where files are saved.

In this section, we discuss threats from a colluding wireless carrier, and list a number of precautions that may help maintain deniability in the presence of such a carrier. Mobile devices are often connected to a cell phone network. It is likely that the wireless carrier maintains activity records, with identifying information and time stamps, of devices interfacing with the network. These records can demonstrate that the device is online and communicating at a given time. The use of the PDE models likely to cause discrepancies between the carrier's logs and the device's standard mode (outer volume) logs. For example, if the carrier has records of a phone call at a given time that occurred when the device was booted in the PDE mode, the device will not have a record of the call in the standard mode. In certain situations, an adversary may be able to collude with the carrier (e.g., a state-based carrier), or compel the carrier to disclose user records (e.g., by court orders). If the user has provided the adversary with

the decoy password, the adversary may find discrepancies between the device logs and the carrier's logs.

This would give the adversary reason to believe that the user has not been completely forth coming. They may then continue to coerce the user for any additional passwords or keys. To restrict the above threats, we provide a list of user practices that must be adhered to when booted in the PDE mode; some of these practices may be onerous to the user. This list shows pitfalls of using PDE systems in practice; however, it is not meant to be comprehensive.

1. When using the device in the PDE mode, it should be left in "Airplane mode" (i.e., antennae off), and the SIM card should be removed. This may prevent the wireless carrier from identifying the device user in their activity logs.

2. A secondary anonymous SIM card should be used, and the phone's identifying information spoofed, if connecting to a mobile network while in the PDE mode (e.g.,IMEI spoofing and software MAC address spoofing). This will restrict any carrier or ISP from directly identifying a suspected device.

3. We strongly discourage the use of mobile data networks in favour of public WiFi hot-spots or Internet pass-through tethering from a PC. Identifying network traffic information (e.g., destination IP address) should be spoofed or obscured with a tool such as Tor3 or a trusted (e.g., employer controlled) VPN when using any type of network connection. This may also restrict an ISP or carrier from correlating the user's behaviour (e.g., if the user is known to frequent certain file hosting service, or news agency).

4. When using the PDE mode, any web services (e.g., email, social networking) should not be used unless a secondary account is created under a pseudonym and is only used in the PDE mode. This will prevent any collusion between the adversary and web service providers with which the user is known to have an account. This includes the device registration account (e.g., Google or iTunes). It is also recommended that auto-backup features (e.g., iCloud or Google Drive) are disabled in the PDE mode.

Here, we evaluate Mobiflage against known attacks and weaknesses.

(a) Password guessing. We rely on the user to choose strong passwords to protect their encryption keys. The current Android encryption pre-boot authentication times-out for 30 seconds after ten failed password attempts. The timeout will slow an online guessing attack, but it may still be feasible, especially when weak passwords are used. An offline dictionary attack is also possible on an image of the device's storage. The adversary does not know the password to derive the offset, but the salt is found in the Android encryption footer. The salt is used with PBKDF2, and is a precaution against pre-generated dictionaries and rainbow tables. The salt cannot be stored at the hidden offset as it is used in the offset calculation. Using the same salt value for both modes enables the adversary to compute one dictionary of candidate keys (after learning the salt), to crack passwords for both modes. Exacerbating the problem is Android's low PBKDF2 iteration count.

On a single core of an Intel i7-2600, at 2000 iterations, we were able to calculate 513.37 ± 1.93 keys per second using the OpenSSL 1.0.1 library. Custom hardware (e.g., FPGA/GPU arrays) and adapted hash implementations can make offline guessing even more efficient. We tested different hash iteration counts in PBKDF2 and found that 200,000 iterations is apparently a fair compromise between security and login delay. On the Intel i7-2600, at 200,000 iterations, we were able to calculate 5.21±0.01 keys per second (i.e., guessing attack becomes 100 times slower). On our Nexus S (1GHz Exynos-3 Cortex-A8) development phone, it required an additional 0.67±0.01 seconds to calculate a single key. Our Motorola Xoom (1GHz Tegra-T20 Cortex-A9) required an additional 0.41 ± 0.001 seconds and an HTC EVO3D (1.2GHz MTM8660 Scorpion) required an additional 0.70 ± 0.01 seconds. This would slow down the boot procedure by approximately two seconds; note that, booting into the PDE mode requires three invocations of PBKDF2: to test the key in the footer, to calculate the offset, and to decrypt the hidden volume key. Possible computational and memory-wise expensive replacements for PBKDF2 can also be used to mitigate custom hardware attacks. In the end, we require users to choose a strong password resilient to guessing.

(b) Cipher issues. An implementation flaw can expose FDE ciphers to a theoretical watermarking attack that has been documented for software such as LUKS [9]. The issue occurs when the disk is sufficiently large and the size of the disk sector index (n)

is small. For example, if n is a 32-bit integer, and there are more than 232 512-byte sectors on the disk, the value of n will eventually roll-over and repeat itself. If the adversary can create a special file with duplicate plaintext blocks at correct locations and convince the user to store the file in their hidden volume, then the adversary can demonstrate the existence of a hidden volume. In the given example, the duplicate plaintext blocks would need to be repeated at 2TB intervals. The adversary will not know what the corresponding ciphertext blocks will be, but finding identical cipher texts spaced at the correct distance would be strong evidence.

This is an implementation issue, and not an issue with the cipher algorithm itself. The problem occurs for all FDE ciphers, including XTS and CBCESSIV, that use a sector index smaller than the total number of disk sectors. To mitigate this problem, a longer integer (e.g., 64-bit) is commonly used for the sector index. We use the 64-bit sector index available in dm-crypt which will not roll over until 8192 Exabytes.

(c) Software issues. Mobiflage seems to effectively isolate the outer and hidden volumes. Apps and files installed in the hidden volumes leave no traces in the outer volume. Android does not use dedicated swap space. When the OS needs more RAM for the foreground app, it does not page entire regions of memory to the disk. Instead, it unloads background apps after copying a small state to the user data partition. For example, the web browser may copy the current URLs of open tabs to disk when unloading, instead of the entire rendered page. When the browser is loaded again, the URL is reloaded. Leakage into swap space and paging files was shown to be an issue for desktop PDE implementations by Czeskis et al. As the outer and hidden user data partitions are isolated from one another in Mobiflage, we do not take any specific measures against leakage through memory paging. The Android Framework is stored in the system partition which is mounted read-only. The Linux kernel is stored in a read-only boot partition which is not mounted onto the OS file system. Leakage through these immutable partitions is also unlikely. Android logs are stored in a RAM buffer, and application logs are stored in the user data partition. Leakage is also unlikely through logs as the user data partitions are isolated and RAM is cleared when the device is powered off. Some devices keep persistent logs at /dev log, for trouble shooting between boots. To prevent leakage through these logs, we mount a tmpfs RAM disk to this mount point when booting into the PDE mode. The logs will remain

persistent between standard mode boots, but no PDE mode logs are kept. Android devices typically have a persistent cache partition used for temporary storage. For example, the Google Play store will download application packages to this partition before installing them on the user data volume. To prevent leakage through the cache partition, we mount a tmpfs RAM disk to cache in the PDE mode; this partition takes 32MB of RAM. An alternative to tmpfs, without sacrificing RAM, is to mount the volume through dm-crypt with a randomly generated one-time key. The key is discarded on reboot, effectively destroying the data on the partition.

(d) Partial storage snapshots. If the adversary has intermittent or regular access to the disk, they may be able to detect modifications to different regions of the disk. If a decoy key has already been divulged, the adversary may surmise the existence of hidden data by correlating file system activities to the changing disk regions. We exclude this possibility assuming the adversary will have access only after acquiring the device from the user, and does not have past snapshots of the storage. If the user is aware that the storage has been imaged (e.g., at a border crossing), they should re-initialize Mobiflage to alter every sector on the disk.

(e) Practical security of multiple hidden volumes. There is some debate over the effectiveness of multiple hidden volumes. Whether or not the user gains any advantage is defined by the scenario. If the user cannot be held indefinitely, and cannot be punished on the suspicion of PDE data alone, she may feign compliance by relinquishing decoy keys. This may be advantageous to the user as, in the absence of indisputable evidence, she will eventually be released. In other scenarios, revealing the existence of one hidden volume may cause the adversary to suspect the existence of additional hidden volumes.

If they can hold the user indefinitely, then they can continue to demand keys. It may in fact hinder the user to reveal any hidden volumes in this situation. However, irrespective of multiple hidden volumes, the adversary can keep punishing a suspect up until the true password is revealed. This is an inherent limitation of PDE schemes and may be alleviated (to some extent) by using a special password to make the hidden data permanently inaccessible.

To understand the performance impact on the regular use of a device, we run severa l

tests on our prototype implementation of Mobiflage. This section summarizes our findings. We use Mobiflage on Nexus S and Motorola Xoom development devices by reading from and writing to the SD card. The command-line tool cp is used to duplicate files on the SD card. We run 20 trials on four files between 50 MB and 200 MB. We evaluate the performance on unencrypted storage, under the default Android encryption, and the Mobiflage scheme. Note that, removable SD storage (as in the Xoom) is apparently much slower than eMMC storage (as in the Nexus S), for all cases. Compared to the unencrypted case, on our Nexus S, Mobiflage reduces IO throughput by almost 10%; in contrast, Android FDE reduces the throughput by 5.5%. On the Motorola Xoom, Mobiflage reduces throughput by 17.6% and Android FDE by 12.6%. Mobiflage seems to decrease throughput by roughly 5% over Android FDE. However, the decreased IO throughput is negligible for regular apps and should not hinder the use of the device. For example, a standard definition 30fps video file may have a combined audio/video bit-rate of 192 KBps. High definition video(e.g., Netflix)is generally below 1024KBps. The reduced speed of Mobiflage (3929 KBps) will still provide adequate buffering to ensure that jitter will not be an issue in these video apps. Note that Blu-ray has a maximum bitrate of 5000 KBps and may cause playback issues, if it is not be attributed to the chosen cipher: XTS requires two AES operations per block; and AES-256 uses fourteen rounds of operations while AES-128 uses ten.

Android apps are first loaded into RAM and do not run directly off the disk. Mobiflage should not affect run time performance of apps. The increase in app load time should also be practically negligible; as of Sept. 2012, the average Android app size is about 6MB, although the size of certain apps (e.g., gaming) is increasing rapidly.

Some hardware, such as the camera, may use direct memory access (DMA) and may be affected: instead of writing directly to the disk, the camera data is processed by the CPU when passing through the dm-crypt layer. We tested the camera on our Nexus S device while in the Mobiflage PDE mode, and did not notice any performance impact.

The required time to encrypt the device is increased on account of the two pass random wipe. The exact time will depend on the size of the external storage partition. Android FDE encrypts external eMMC partitions in-place. As such, Mobiflage will take twice

as longto encrypt these partitions. Removable SD cards are not encrypted by Android FDE, so we cannot provide a static comparison. Our Nexus S has only 1GB internal, and 15GB eMMC external storage. After three initializations, we found that on average the default Android FDE require done hour and five minutes, and Mobiflage required just under two hours. The Motorola Xoom required one hour and fifteen minutes on average for the default Android FDE to encrypt the 32GB internal storage. Encrypting with Mobiflage required an additional 73 minutes when used with a 8GB SanDisk SD card. Power consumption will likely be increased for disk activity. This problem is inherent to all FDE, and is not unique to Mobiflage. Background processes that have high IO activity should be disabled, or IO should be buffered and batched to reduce power consumption.

Encryption is the process of encoding user data on an Android device using an encrypted key. Once a device is encrypted, all user-created data is automatically encrypted before committing it to disk and all reads automatically decrypt data before returning it to the calling process. Android is a modern mobile platform that was designed to be truly open. Android applications make use of advanced hardware and software, as well as local and served data, exposed through the platform to bring innovation and value to consumers. To protect that value, the platform must offer an application environment that ensures the security of users, data, applications, the device, and the network.

Securing an open platform requires a robust security architecture and rigorous security programs. Android was designed with multi-layered security that provides the flexibility required for an open platform, while providing protection for all users of the platform.

Android was designed with developers in mind. Security controls were designed to reduce the burden on developers. Security-savvy developers can easily work with and rely on flexible security controls. Developers less familiar with security will be protected by safe defaults.

Android was designed with device users in mind. Users are provided visibility into how applications work, and control over those applications. This design includes the expectation that attackers would attempt to perform common attacks, such as social

engineering attacks to convince device users to install malware, and attacks on third-party applications on Android. Android was designed to both reduce the probability of these attacks and greatly limit the impact of the attack in the event it was successful.

Cryptography technique needs some algorithm for encryption of data. The encryption/decryption process refers to the operation of dividing and replacing an arrangement of the original image .the image can be decomposed into blocks; each one contains a specific number of pixels. The blocks are transferred into new locations. For better process the block size should be small, because fewer pixels keep the neighbours. In this case the correlation will be decreased and thus it becomes difficult to predict the value of any given pixel from the values of its neighbours. At the receiver side, the original image can be obtained by the inverse transformation of the blocks.

Based on the assessment criteria defined above, Android's encryption systems are discussed and assessed in this section to derive potential attack scenarios. Android offers two primary encryption systems: First a file-system based encryption system that needs to be activated by the user administrator, and second the Android Key Chain, which can be employed by the developer to store credentials used in an application in a secure way on the file-system. Apart from these encryption systems, the following analysis also discusses the various backup facilities on the Android platform, and cloud-storage components that directly or indirectly influence the security of data protected by the encryption systems. Although the backup system does not fall into the category of encryption systems, the protection mechanisms (or their lack) are crucial when deploying Android in security critical scenarios. An attacker who might not be able to break the device encryption system could still gain access to data by gaining access to backup files on a laptop or by breaking into the user's Google account.

Dm-crypt the primary encryption system on Android is a file-system based encryption system based on dm-crypt3, which has been available in Linux kernels since Version 2.6.x. The Android file-system encryption has been introduced in Android 3.0, which was solely used on tablets. This version has later been adapted for smartphones and made available for those devices as Version 4.0. In Version 4.4, the employed KDF has been changed, which improves the security level of the used passcodes.

An overview of the encryption system is given in Figure2.3. The user is required to enter the passcode as the first step in the Android boot process. The KDF is then used to derive a symmetric key from the user's passcode. This symmetric key protects the actual file encryption master key which is at the top of the key hierarchy required for the file-encryption system. The encryption system must be activated manually by the user, or be enforced by the corresponding MDM rule. The activation of the system requires the user to set an encryption passcode, which is also used for the phone's lock screen. When activating the encryption system, it is mandatory to activate the passcode screen lock functionality. Other lock-screens, such as the pattern lock functionality of the face-unlock system cannot be used when file-system encryption is activated.

1) Assessment: Regarding the first assessment criterion (key derivation function), a distinction between Android 3.0 to 4.3, and 4.4 must be made. While 3.0 to 4.3 use the Password Based-Key-Derivation-Function 2 (PBKDF2) [9] with a SHA1 based HMAC to derive a symmetric key from the user's passcode, Android 4.4. Employs the SCRYPT function [10], which complicates parallel brute-force attacks by requiring computational and memory intensive resources.

Android 3.0 to 4.3: The PBKDF2 derivation process employs 2000 iterations to slow down possible brute-force attacks. The input values for the key derivation process are the user's passcode and a random salt value (16 bytes). The salt value is randomly generated during the activation of the encryption process and eliminates the possibility for an attacker to pre-calculate derived keys in order to speed-up the attack.

Android 4.4: The default parameters used for the SCRYPT function for the dm-crypt system are N = 15, r = 3, p = 1, where N influences the required CPU resources, r defines the memory requirements and p influences the parallelization cost. This setting can be overwritten during the Android build process by defining specific parameters in the build. Properties file. Since these parameters can therefore be adapted by the different device manufacturers, this flexibility also results in different brute-force times and in different passcode properties that assure a secure encryption environment. Regarding the second assessment criterion (configuration capabilities), it has to be stated that the Android file-system encryption is not activated per default, and must either be activated manually by the user enforced by the respective MDM rule. Apart

from the activation of the encryption system, the appropriate rules4 for defining passcode properties must also be set by the administrator according to the security requirements of the envisaged deployment scenario and the times required to carry out a possible brute-force attack. From a developer's point of view, it is not possible to influence the configuration of the encryption system. Due to the requirement to manually activate the system, the developer cannot assume that the Android system used to run the application is protected by the file-system encryption system. This is especially important for BYOD scenarios, where the security of the system depends on the settings chosen by the user and cannot be enforced by MDM.

2) Attack Scenarios: From the obtained assessment results, several attack scenarios can be derived for Android's file-based encryption system. The first attack scenario is based on the fact that Android's encryption system must be activated by the user manually or be enforced by the corresponding MDM rule. In addition, the security of the system depends on the strength of the passcode. Thus, the negligence to activate the system or to choose weak passcode properties allow for attacks on the encryption system when the device is stolen.

To understand the performance impact on the regular use of a device, we run severa l tests on our prototype implementation of Mobiflage. This section summarizes our findings. We use Mobiflage on Nexus S and Motorola Xoom development devices by reading from and writing to the SD card. The command-line tool cp is used to duplicate files on the SD card. We run 20 trials on four files between 50 MB and 200 MB. We evaluate the performance on unencrypted storage, under the default Android encryption, and the Mobiflage scheme. Note that, removable SD storage (as in the Xoom) is apparently much slower than eMMC storage (as in the Nexus S), for all cases. Compared to the unencrypted case, on our Nexus S, Mobiflage reduces IO throughput by almost 10%; in contrast, Android FDE reduces the throughput by 5.5%. On the Motorola Xoom, Mobiflage reduces throughput by 17.6% and Android FDE by 12.6%. Mobiflage seems to decrease throughput by roughly 5% over Android FDE. However, the decreased IO throughput is negligible for regular apps and should not hinder the use of the device. For example, a standard definition 30fps video file may have a combined audio/video bit-rate of 192 KBps. High definition video(e.g., Netflix)is generally below 1024KBps. The reduced speed of Mobiflage (3929 KBps)

will still provide adequate buffering to ensure that jitter will not be an issue in these video apps. Note that Blu-ray has a maximum bitrate of 5000 KBps and may cause playback issues, if it is not be attributed to the chosen cipher: XTS requires two AES operations per block; and AES-256 uses fourteen rounds of operations while AES-128 uses ten.Another possible attack scenario targets the encryption system's KDF. The security of Android's file-system encryption system primarily depends on the length and the properties of the user's passcode used to derive the symmetric key, which protects the actual file-system encryption keys. Android uses a software-only architecture for encrypting the file-system. Thus, the system is susceptible to external brute-force attacks, where computationally expensive operations are outsourced to powerful computing units. For concrete numbers, we again need to distinguish between different Android versions.

Android 3.0 to 4.3: Since the KDF involves standard hash algorithms, optimized implementations can be leveraged to speed up the required calculations. In order to attach a price tag to brute-force attacks carried out on Android, we have used the pricing model of the Amazon EC2 cloud computing instances to calculate the costs for carrying out attacks on passcodes with different length and properties. Thereby, two scenarios have been evaluated: While the first scenario utilizes calculations based on standard CPUs available within Amazon cloud instances, the second scenario considers special Amazon instances that employ GPUs that speed up the hash calculation process.. In this table, only the prices for the on-demand instances are considered, which – in contrast to the reserved instances – do not require any upfront payments. The table shows the brute-force times for different passcode lengths and properties. Password complexities were arranged.

For the GPU scenario, the brute-force time calculation is based on a special Amazon instance that employs two NVIDIA Tesla Fermi M2050 GPUs. Although we did not implement a tool that carries out the brute-force attack on these GPUs, a good estimation on the required time can be provided by taking a closer look at the PBKDF2 function and the number of required hash operations and the block size of the input data.

In paper 1, we have studied the basic terms used in cryptography, its purpose and compared the encryption techniques used in cryptography. The implementation of best algorithm found, has been depicted using web based and android based implementation.

In paper 2, the issues related to privacy have been addressed along with a proposal of a new implementation of cloud computing architecture. We have used segmentation technique to apply different encryption algorithms on different segments on the basis of odd even index values of segments.

Cloud computing builds on decades of research in virtualization, distributed computing, utility computing, and more recently networking, web and software service& It implies a service oriented architecture, reduced information technology overhead for the end-user, great flexibility, reduced total cost of ownership, on demand services and many other thing.

Cloud computing is one of the fastest growing technologies in the field of computation and storage of data and files. Cloud computing refers to manipulating, configuring and accessing the resources remotely. It offers online data storage, infrastructure and application on demand of the client with minimal management effort. It is a kind of Internet-based computing that provides shared processing resources and data to computers and other devices on demand.

There are three basic services models –

- infrastructure as a service (IaaS),
- platform as a service (PaaS) and
- Software as a service (SaaS), working behind the scene making the cloud computing feasible and accessible to the users.

There are several attributes of cloud computing which allows users to deploy the cloud services. These are as follows:

1) ***Reliability***: Cloud computing is a reliable and consistent option when there is a managed service platform.

2) ***Scalability***: It provides the ability to scale the bandwidth and storage space of the system.

3) ***Elasticity***: Users can rapidly increase and decrease their computing resources as needed.

4) ***On-demand service***: Users are provided the services according to their demand and are allowed to pay for only the resources they actually use for a particular time frame.

Along with all the advantages of the cloud services, the main area of concern in the field of cloud computing is security. Cloud Computing stores the data in the open environment at external servers which makes the data prone to potential attacks. The data owners feel unsafe to store their data on cloud. The cloud provider should ensure that the architecture and the infrastructure used by them is secure and the data and applications of the client are not compromised. Even at the time of transmission of data it is required that the data is not prone to any kind of active or passive attack.

These records can demonstrate that the device is online and communicating at a given time. The use of the PDE models likely to cause discrepancies between the carrier's logs and the device's standard mode (outer volume) logs. For example, if the carrier has records of a phone call at a given time that occurred when the device was booted in the PDE mode, the device will not have a record of the call in the standard mode. In certain situations, an adversary may be able to collude with the carrier (e.g., a state-based carrier), or compel the carrier to disclose user records (e.g., by court orders). If the user has provided the adversary with the decoy password, the adversary may find discrepancies between the device logs and the carrier's logs Data Security is a crucial element that warrants scrutiny.

Enterprises are reluctant to buy an assurance of business data security from vendors. They fear losing data to competition and the data confidentiality of consumers. In many instances, the actual storage location is not disclosed, adding onto the security concerns of enterprises. In the existing models, firewalls across data centers (owned by enterprises) protect this sensitive information. In the cloud model, Service providers are responsible for maintaining data security and enterprises would have to rely on them.

All business applications have Service level agreements that are stringently followed. Operational teams play a key role in management of service level agreements and runtime governance of applications. In production environments, operational teams support

In this section, we discuss threats from a colluding wireless carrier, and list a number of precautions that may help maintain deniability in the presence of such a carrier. Mobile devices are often connected to a cell phone network. It is likely that the wireless carrier maintains activity records, with identifying information and time stamps, of devices interfacing with the network.

These records can demonstrate that the device is online and communicating at a given time. The use of the PDE models likely to cause discrepancies between the carrier's logs and the device's standard mode (outer volume) logs. For example, if the carrier has records of a phone call at a given time that occurred when the device was booted in the PDE mode, the device will not have a record of the call in the standard mode. In certain situations, an adversary may be able to collude with the carrier (e.g., a state-based carrier), or compel the carrier to disclose user records (e.g., by court orders). If the user has provided the adversary with the decoy password, the adversary may find discrepancies between the device logs and the carrier's logs.

This would give the adversary reason to believe that the user has not been completely forth coming. They may then continue to coerce the user for any additional passwords or keys. To restrict the above threats, we provide a list of user practices that must be adhered to when booted in the PDE mode; some of these practices may be onerous to the user. This list shows pitfalls of using PDE systems in practice; however, it is not meant to be comprehensive. All business applications have Service level agreements that are stringently followed. Operational teams play a key role in management of service level agreements and runtime governance of applications. In production environments, operational teams support

CHAPTER 6

SNAPSHOTS OF IMPLEMENTATION

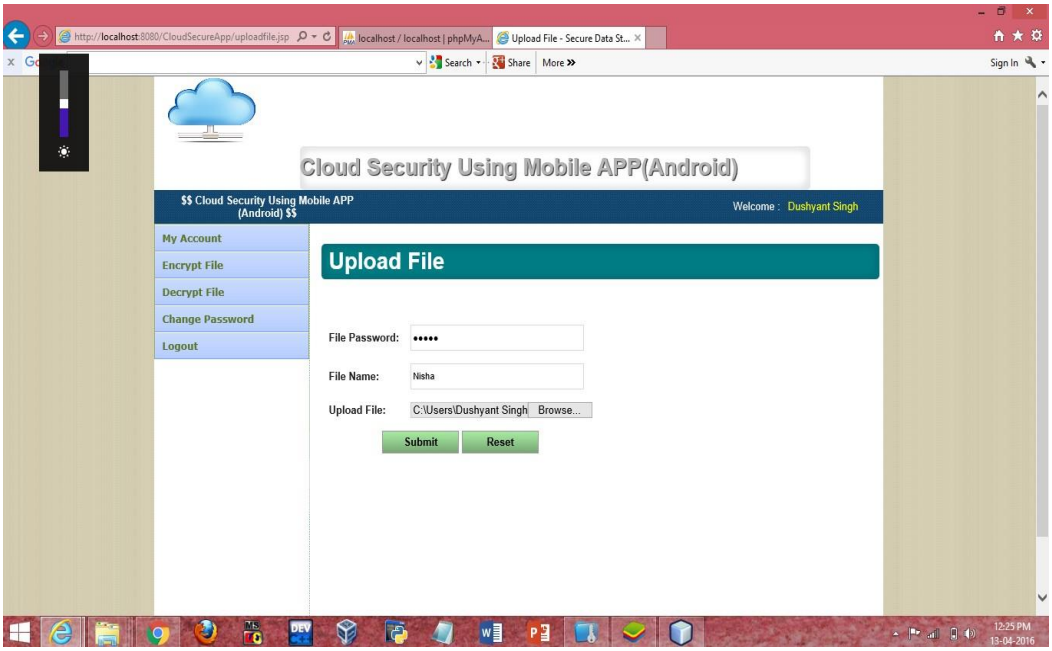


Fig. 6.1 Upload File Page

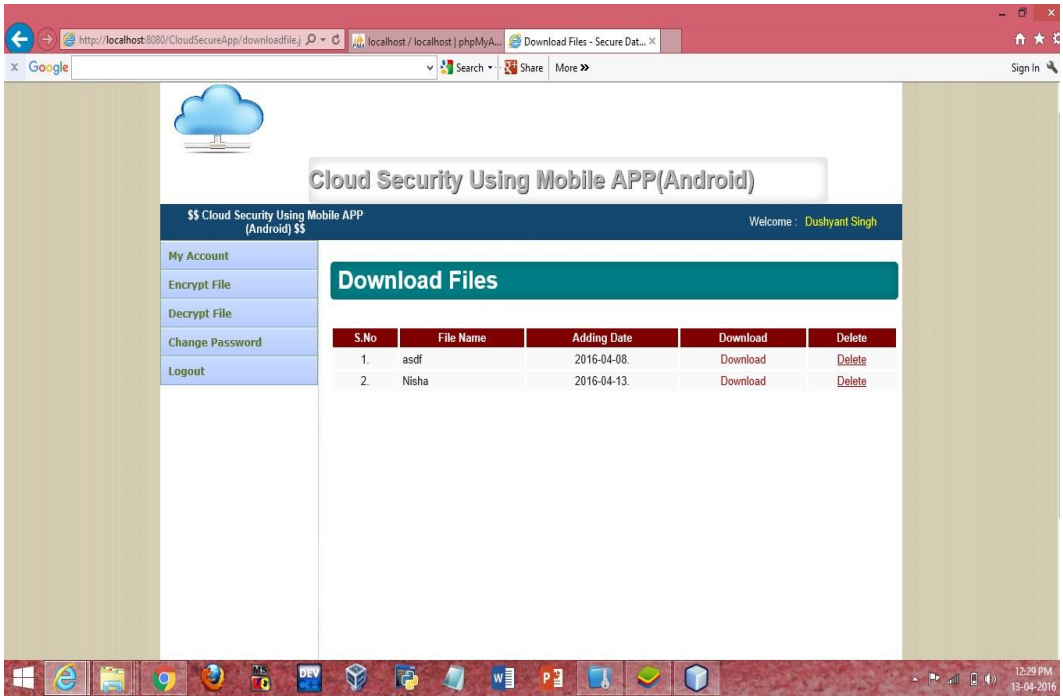


Fig. 6.2 List of Files Uploaded

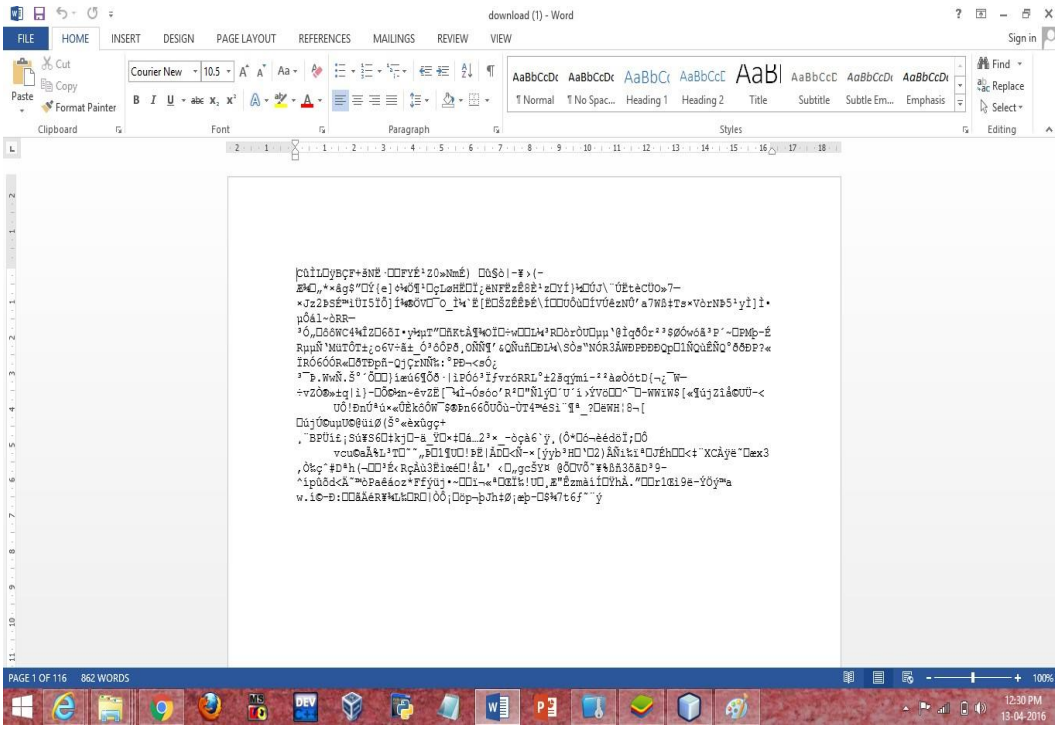


Fig. 6.3 Opened Encrypted File

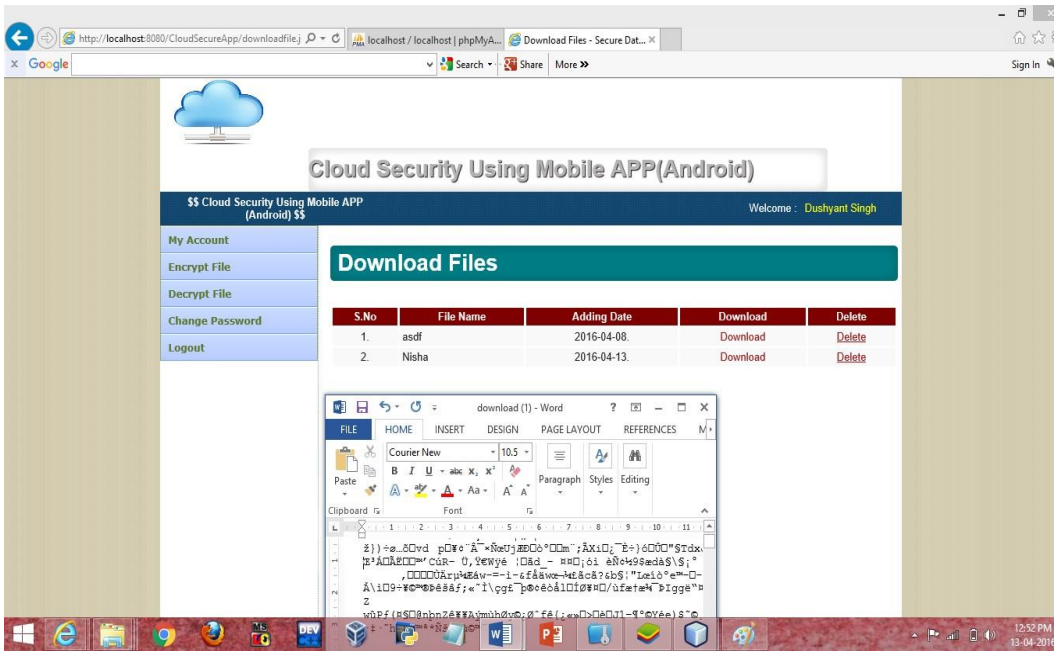


Fig. 6.4 Download Encrypted File

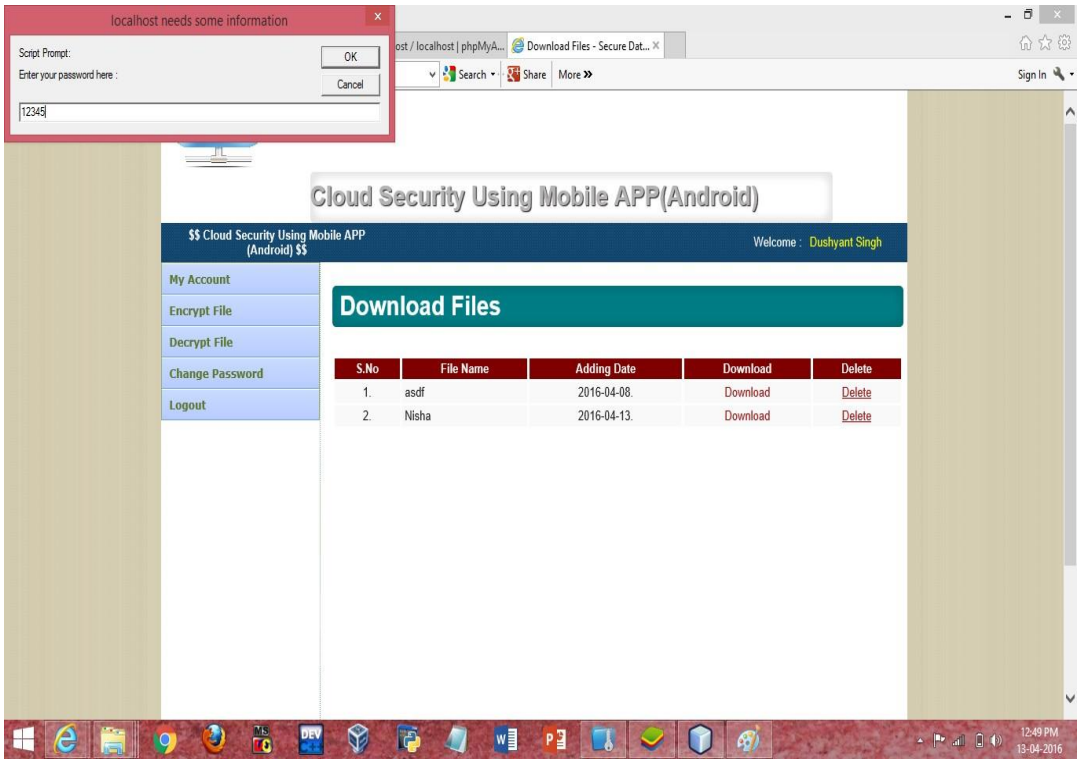


Fig. 6.5 Encrypt File Password

CHAPTER 7

CONCLUSION AND FUTURE WORK

In research paper 1, we have analysed DES, AES RIJNDAEL, AES SERPENT, and BLOWFISH closely to find the best algorithm to be used. Based on our study we concluded that AES RIJNDAEL is the best encryption algorithm that can be implemented on cloud. We have made an application in java using NETBEANS IDE that implements this algorithm on files for storage on cloud and its subsequent retrieval from the cloud. We have also made an app for the same so that the same can be accessed from web as well as mobile. We have used ECLIPSE IDE for making our android app. To depict the implementation of cloud we are using WAMP server. We have saved our SQL query in it. For future work, to increase security we can break the file in to parts and implement AES RIJNDAEL on first part and AES SERPENT on second part of the file. This will enhance the level of security in our application. We have one more approach to increase the security of file. We can apply two different encryption algorithms on same file based on its size. If the size of file is even we can apply AES RIJNDAEL else AES SERPENT. The outsider would never be able to know what algorithm might have been applied, since he is unaware of the size of the file uploaded on cloud.

In research paper 2, we presented a new approach which provides security for data outsourced at CSP. By employing the dual cryptography after segmentation at the user side, we protect outsourced data from outsider's attack. Since, DO stores its data at CSP in encrypted form and, keys are known only to DO and respected users group, data confidentiality is ensured. Segmentation provides an extra edge of security. To ensure easy retrieval of outsourced data, the scheme has used index-hash mapping table. Public key cryptography and MD5 ensure the entity authentication and data integrity respectively. The concept of multi-cloud provides additional security to the data stored at CSP. Within the CSP the data is dynamically shuffled both inter-cloud and intra-cloud. Public key cryptography with segmentation has protected the data from outsiders in our approach. The security has improved highly in the proposed scheme.

Since data now a days, is being stored and retrieved digitally through electronic means from any geographical coordinate, our objective is to implement a robust mechanism that assures both its security and integrity comprehensively.

Our objective is that this application should fortify the users to exchange files effectively, in a secured manner, ensuring the confidentiality of communication. It may be utilized in classified communications like criminal investigations, business communications, and the like. The communication and data should not just be encoded over the transmission channels and cloud but the exclusive feature of Device Obscurity should ensure the content security at the user's device / handset level also.

The reliable and fail-safe cloud service should not only guarantee the integrity of the stored information but also its security because the multimedia content shall be stored in an encoded manner plus every user's data is enveloped in his/her own individual user account on the cloud.

We propose to provide convenience to the user through simple, understandable yet secure communication interfaces that are not just easily navigable but also interact intelligently with the user. Data that is protected at the cloud level too so that if there is any unauthorized access, the illegible user is unable to decipher the exact content. Our goal is to shield the retrieval of user data using multi-layer virtual encryption loops to minimize illegible access to the negligible levels.

We propose to develop an application that utilizes the latest possible tools, technologies and techniques available in the market and provides a simplified solution to the end user that not only protects the data but is also smart, intelligent, consumes less resources and efficiently reliable.

Through this application, we propose to impart confidence to the end user in relation to the security and integrity of his/her confidential information that is of great deal of importance so that when he/she operates on our application, not only the intelligent, comprehensive mechanisms at all points of data transmission protect the user data but also make the life of people miserable who even think of making an attempt to eavesdrop it.

After studying the prevalent researches, we concluded that:

Most of the studies were based on the fact that they developed their own product in order to secure one area but not the entire content transmission as a whole. All the

studies focused on one idea only and they did not concentrate on eliminating its disadvantages whereas we attempt to balance the demerits of one with the merits of another technique.

The studies mostly depend on the complicated calculations in order to secure the storage of data whereas we shall focus on simplistic calculations that not only use minimum resources but also produce efficient output in the minimum possible turnaround time.

The studies clearly show that they do not protect the classified data of the user which might be consisting of extra sensitive information that is useful for some investigation purpose or some verification purpose, right from the point of its creation and delivery whereas in our research, we make a conscious attempt to protect the data at all the levels i.e. Device, Network & Cloud because we believe that data is protected comprehensively once it become non – accessible to the outside world from the different dimensions.

The techniques do not discuss about the efficient resource utilization of the user's resources be it hardware or software whereas in our study we plug and bridge this gap through minimizing the consumption using simple yet efficient obscurity and encoding mechanisms that put minimum possible load. The studies showed a large gap in harnessing the power of latest technologies available in the market to its full potential whereas in our study we shall make an attempt to use the latest tools, techniques and technologies that only are output – oriented but also possess a vast future scope of expansion possibilities.

ABBREVIATIONS

SQL	Structured Query Language
ISP	Internet Service Provider
AES	Advanced Encryption Algorithm
NIST	National Institute of Standards and Technology
AES	Advanced Encryption Standard
DES	Data Encryption Standard
MDM	Mobile Device Management
OS	Operating System
RAM	Random Access Memory
GUI	Graphical User Interface
GB	Gigabytes
MB	Megabyte

REFERENCES

[1]. Tania Gaur and Divya sharma (2016) on A Secure and Efficient Client-Side Encryption Scheme in Cloud Computing.

[2]. Abhishek Mohta, Ravi Kant Sahu and Lalit Kumar Awasthi (2012) on Robust Data Security for Cloud while using Third Party Auditor, *International Journal of Advanced Research in Computer Science and Software Engineering ISSN: 2277 128X.*

[3]. Dr. Chander Kant and Yogesh sharma (2013) on Enhanced Security Architecture for Cloud Data Security, *International Journal of Advanced Research in Computer Science and Sofware Engineering ISSN: 2277 128X.*

[4]. Surendra Singh Rathod and Anand Rajawat (2015) on The Research on Cloud Server Storage Security Using TPA, *International Journal of Advanced Research in Computer Science and Software Engineering ISSN: 2277 128X.*

[5]. Ankit R. Mune1 and P. R. Pardhi (2015) on Security for cloud computing data using a security cloud as a Third party auditor (TPA): A Survey, *International Journal of Advanced Research in Computer and Communication Engineering ISSN: 2278-1021.*

[6]. Hasan Omar Al-Sakran (2015) on Accessing Secured Data In Cloud Computing Environment, *International Journal of Network Security & Its Applications (IJNSA).*

[7]. Sunita Bhati1, Anita Bhati and S. K. Sharma on A New Approach towards Encryption Schemes: Byte – Rotation Encryption Algorithm, *Proceedings of the World Congress on Engineering and Computer Science.*

[8]. Prakash Sawle and Trupti Baraskar (2016) on Survey on data classification and data encryption techniques used in cloud computing, *International Journal of Computer Applications.*

[9]. Shakeeba S. Khan and Prof. R.R. Tuteja (2015) on Security in Cloud Computing using Cryptographic Algorithms, *International Journal of Innovative Research in Computer and Communication Engineering ISSN: 220-9801*.

[10]. Parsi Kalpana and Sudha Singaraju (2012) on Data Security in Cloud Computing using RSA Algorithm, *International Journal of Research in Computer and Communication Technology(IJRCCT) ISSN: 2278-5841*.

[11]. Neha Tirthani and Ganesan R on Data Security in Cloud Architecture Based on Diffie Hellman and Elliptical Curve Cryptography.

[12]. Saranyak, Mohanapriya R and Udhayan J on A Review on Symmetric Key Encryption Techniquesin Cryptography, *International Journal of Science, Engineering and Technology Research (IJSETR).*

[13]. Neha Tirthani and Ganesan R on Data Security in Cloud Architecture Based on Diffie Hellman and Elliptical Curve Cryptography.

[14]. Saranyak, Mohanapriya R and Udhayan J on A Review on Symmetric Key Encryption Techniquesin Cryptography, *International Journal of Science, Engineering and Technology Research (IJSETR).*

[15]. Mary Cindy Ah Kioon, ZhaoShun Wang and Shubra Deb Das on Security Analysis of MD5 algorithm in Password Storage .

[16]. V. Tharun Deep and Dr. Venkata Siva Reddy on Comparative Analysis of Aes Finalist Algorithms And Low Power Methodology for Rc6 Block Cipher - *A Review, International Journal For Technological Research In Engineering ISSN: 2347 – 4718.*