UNIVERSITY OF CAPE TOWN

DEPARTMENT OF ELECTRICAL ENGINEERING

DATA STORAGE SECURITY FOR CLOUD COMPUTING USING ELLIPTIC CURVE CRYPTOGRAPHY.

GEORGE ONYANGO BUOP

Supervisor: DR. ALEXANDRU MURGU

A thesis Presented for the Degree of

Master of Engineering in Telecommunication

Declaration

I declare that all the work in the document, save for that which is properly acknowledged, is my own. I hereby declare that I have not submitted this material, either in whole or in part, for a degree at this or any other institution.

Signature …Signed by candidate……

Cape Town

Acknowledgement

First and foremost, my appreciation goes to the Almighty God who sustained me throughout the period of this program. I also want to acknowledge the contributions of my supervisor, Dr Alexandru Murgu, for the comments, corrections, suggestions and all other forms of support I have received towards the completion of this research work.

My sincere appreciation is extended to colleagues who have been of great encouragement through their technical skills and suggestions while going through this path. I also would like to appreciate the friends who made the foreign land a home away from home, through their company, their support and continuous encouragement. Finally, my special appreciation goes to my parents and my family for being an inspiration, for the sacrifices they made to ensure I was able to go through this program to completion and for being there for me throughout this period. Thank you all.

Table of Contents

List of Figures

List of Tables

List of Symbols

∞	Point at infinity
C_m	Ciphertext of message
Ds	Digital signature
E	An elliptic curve
$E(\mathbb{F}_{2^m})$	An elliptic curve E defined over a finite field $\mathbb{F}_{2^m}$
$E(\mathbb{F}_p)$	An elliptic curve E defined over a finite field $\mathbb{F}_p$
H	Hash function
ID_U	User identity
M	Message unit
Mk	Master Private Key
n	number of elements in group (order of group)
p	Prime number
P	A point on the elliptic curve
Pr_U	User private key
PU_U	User public key
$\mathbb{Z}$	Set of integer
$\mathbb{F}_p$	Finite field with element, where p is a prime number
$\mathbb{G}$	Group

Abbreviations

AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	Application Programming Interfaces
CA	Certification Authority
CSP	Cloud Service Provider
CSS	Cascading Style Sheet
DLP	Discrete Logarithm Problem
DOS	Denial of Service
DSA	Digital Signature Algorithm
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman Key Exchange
ECDLP	Elliptic Curve Discrete Logarithm Problem
ECDSA	Elliptic Curve Digital Signature Algorithm
ECIES	Elliptic Curve Integrated Encryption Scheme
GUI	Graphical User Interface
HIBC	Hierarchical Identity-Based Cryptography
HMAC	Hashed Message Authentication Code
IaaS	Infrastructure as a Service
IBC	Identity Based Cryptography
IBE	Identity Based Encryption

IBS	Identity Based Signature
IEEE	Institute of Electrical and Electronics Engineers
ISO/IEC	International Organization for Standardization/ International Electrotechnical Commission
KDF	Key Derivation Function
MAC	Message Authentication Code
MySQL	My Software Query Language
NIST	National Institute of Standards and Technology
OS	Operating System
PaaS	Platform as a Service
PCs	Personal Computers
PKG	Private Key Generator
PKI	Public Key Infrastructure
QoS	Quality of Service
RA	Registration Authority
RSA	Rivest-Shamir-Adleman
SaaS	Software as a Service
SHA	Secure Hash Algorithm
Sk	Shared Secret Key
SP	Service Provider
TA	Trusted Authority
TC	Trusted Cloud

Abstract

Institutions and enterprises are moving towards more service availability, managed risk and at the same time, aim at reducing cost. Cloud Computing is a growing technology, thriving in the fields of information communication and data storage. With the proliferation of online activity, more and more information is saved as data every day. This means that more data is being stored in the cloud than ever before. Data that is stored online often holds private information – such as addresses, payment details and medical documentation. These become the target of cyber criminals. There is therefore growing need to protect these data from threats and issues such as data breach and leakage, data loss, account takeover or hijackings, among others.

Cryptography refers to securing the information and communication techniques based on mathematical concepts and algorithms which transform messages in ways that are hard to decipher. Cryptography is one of the techniques we could protect data stored in the cloud as it enables security properties of data confidentiality and integrity.

This research investigates the security issues that affect storage of data in the cloud. This thesis also discusses the previous research work and the currently available technology and techniques that are used for securing data in the cloud. This thesis then presents a novel scheme for security of data stored in Cloud Computing by using Elliptic Curve Integrated Encryption Scheme (ECIES) that provides for confidentiality and integrity. This scheme also uses Identity Based Cryptography (IBC) for more efficient key management. The proposed scheme combines the security of Identity-Based Cryptography (IBC), Trusted cloud (TC), and Elliptic Curve Cryptography (ECC) to reduce system complexity and provide more security for cloud computing applications. The research shows that it is possible to securely store confidential user data on a Public Cloud such as Amazon S3 or Windows Azure Storage without the need to trust the Cloud Provider and with minimal overhead in processing time.

The results of implementing the proposed scheme shows faster and more efficient communication operation when it comes to key generation as well as encryption and decryption. The difference in the time taken for these operations is as a result of the use of ECC algorithm which has a small key size and hence highly efficient compared with other types of asymmetric cryptography. The results obtained show the scheme is more efficient, when compared with other classification techniques in the literature.

Chapter 1. GENERAL INTRODUCTION

1.1 What is Cloud Computing

The management of data, which is a valuable resource for organizations and individuals, is a very important task. For a long time, storage of data has been done using computer hardware such as hard discs, DVDs, CDs, discs, and floppy discs. With the advancements in database systems, networking and the internet, new computing models have come up, including Cloud Computing (CC). [1].

According to the U.S. National Institute of Standards and Technology (NIST), Cloud Computing is a model for convenient, on-demand, network access to a shared pool of configurable computing resources such as networks, servers, storage, applications, and services, that can be rapidly provisioned and released with minimal management effort or service provider interaction. [2]. In CC, data and applications are maintained with the use of central remote servers and the internet. It allows the consumers/users to access the cloud services wherever they are, at whatever time, in whichever way, on-demand or as per the pay per use principle. In their efforts to identify new and better methods to drive their businesses, enterprises have shifted to solutions that provide lower cost in computing systems, resulting in the exponential growth of CC.

1.1.1 Characteristics of Cloud Computing

It is the characteristics of CC that differentiate it from the other computing models. These characteristics include:

- *Broad network access*: This describes the availability of the computing capabilities to users over the network. Using standard mechanisms, users are able to access cloud resources through various heterogeneous platforms e.g. by using laptops, tablets, mobile phones etc.
- *On-demand self-service*: This refers to the ability of a consumer to provision computing capabilities automatically as and when required, without the human interaction with each service provider.

- *Resource pooling*: Multiple consumers are served with the providers' pooled computing resources using a model, with different virtual and physical resources dynamically assigned and reassigned depending on the demand of consumer
- *Rapid elasticity*: This refers to the rapid and elastic provision of computing capabilities to quickly scale out, and rapid release to quickly scale in. The capabilities that are provisioned to consumers are (from the user's point of view) unlimited and can be purchased in any quantity at any time. Elasticity increases service capacity during busy periods, and reduces capacity during customers' off-peak periods, enabling cloud consumers to minimize costs while meeting their service quality expectations
- *Measured service:* As the service is provided under the pay-as-you-use business model, the usage of services and resources can be metered and automatically billed for each particular user session.

1.1.2 Cloud Computing Architecture

The cloud computing architecture can be categorized into two, the front end and the back end. The front end is the client side that consists of applications such as web browsers, and interfaces used by the users to access the CC platforms.

The backend alludes to the cloud itself. It consists of all the resource that are responsible for the CC services. It includes huge data storage, virtual machines, security mechanism, services, deployment models, servers etc. The front end and back end are connected through a network, normally through the internet.

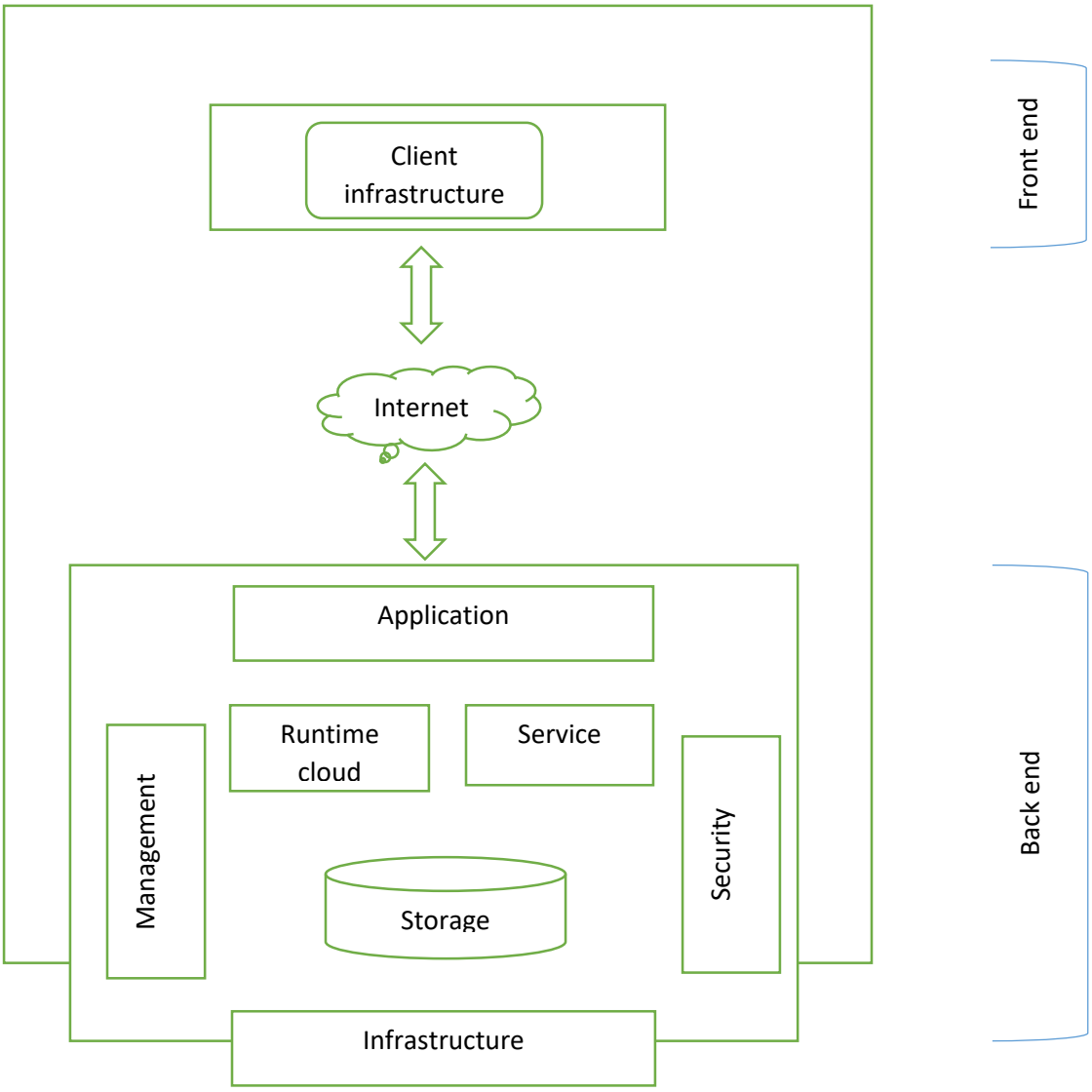


Figure 1.1 Graphical view of Cloud Computing Architecture [52].

1.1.3 Cloud Computing Service Delivery Models

Cloud computing offers a highly flexible and scalable IT environment while most of the maintenance, implementation and management of the infrastructure is performed by Cloud Service Providers (CSPs). There are several types of services and resources that can be provided by CSPs. The most common services can be classified into three: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS).

Software as a Service (SaaS)

Software as a Service (SaaS) solutions deliver software applications over the Web. A SaaS provider deploys software to the user on demand, commonly through a licensing model. The provider may host the application on its own server infrastructure or use another vendor's hardware [3]. Usually, the expense of this service depends on how many users and how long they use the service. The service enables customers to reduce hardware and software resources. Google Docs is an example of SaaS model.

Platform as a Service (PaaS)

Platform as a Service (PaaS) gives consumers the ability to deploy onto a cloud infrastructure consumer-created or consumer-acquired applications, which were created using programming languages and tools supported by the CSPs. Using the concept of PaaS, software developers can build Web applications without having to install the software building tools on their own computer, and then distribute or deploy their apps to the cloud easily. The CSP manages resources at the hardware level, where the hardware requirements, such as CPU, memory and network, will be allocated to the rented platform when they are needed by the customer. The PaaS model provides a lower cost of entry for application designers and distributors, by supporting the complete software development life cycle (SDLC) of the Web app, thereby eliminating the need for the acquisition of hardware and software resources [3]. An example of this type of service is the Google App engine.

Infrastructure as a Service (IaaS)

This is the service model that involves the provision of computing resources and services on which users can deploy and run their own software, including applications and operating systems. These resources are flexible in the sense that they can be scaled up or down as required by the customer. This type of service enables a customer to deploy an IT infrastructure without the expensive start-up cost required. IaaS allows cloud Consumers to run any operating systems and applications of their choice on the hardware and resource abstraction layer (hypervisors) furnished by the cloud provider. A Consumer's operating systems and applications can be migrated to the cloud Provider's hardware, potentially replacing a company's data center infrastructure [4].

1.1.4 Cloud Computing Deployment Models

There are four models for deployment of Cloud computing, namely, Public, Private, Hybrid and Community. The models are classified according to who owns and who uses the service. The choice of a model therefore depends on the organization's objectives and business needs.

Private Cloud

The cloud computing services and infrastructure are owned and served by an organization. A private cloud is used to maximize the utilization of computing resources within the organization. Organizations with high security and privacy concerns would therefore prefer the private cloud model option over other cloud models that involve sharing resources with other organizations. Furthermore, organizations with mission-critical applications may prefer to rely on their own ability to manage and control their in-house infrastructure [5].

Public Cloud

Cloud services are available to the general public and are managed by CC service providers. The providers own and manage their cloud infrastructures. A public cloud has multi-tenant capabilities, and is shared by a large number of customers who have nothing or very little in common. It is the most used deployment model. The nature of the public cloud is that you only pay for what you use and is therefore more cost efficient.

Community Cloud

In community clouds, resources are available for a number of individuals or groups who have shared interests, or common requirements and applications, unlike in public cloud in which users do not have shared common interests. Community cloud may be overseen, managed and worked by a third party or one of organizations in the community or some consolidation of them. The community cloud adds more cost-effective value than the private cloud. This is because the cost is shared among the organizations involved.

Hybrid Cloud

A hybrid cloud combines several deployment models. There is a management framework that ensures that the environments appear as a single cloud. Adoption of hybrid cloud may be needed due to the strong requirements for security, price and performance. The use of more than one model is aimed at harnessing the benefits of the different models used in the hybrid model.

1.1.5 Advantages and Limitations of Cloud Computing

This section will explain the major advantages and the limitations that come with cloud computing [5].

Advantages of Cloud Computing

- Availability: Cloud computing customers can access their resources at any time through a standard internet connection.
- Elasticity: Customer in cloud computing can increase or decrease their computing resources transparently and rapidly as needed, and release the resources which are not required to other users.
- Cost savings: By using the pay per use model, cloud computing customers pay for only the resources which they need. The cost saving is the most interesting thing for the companies when they are evaluating cloud computing.

- Mobility: By using cloud computing, customers can access the cloud services from any mobile devices that are connected to the internet (such as laptop, smart phone, tablet, etc.), and from any location in the world.
- Scalability: Customers of cloud can scale the resources which they use depending on their demand. For example, instead of purchasing additional hardware the customer can rent from cloud infrastructure in less cost.
- Unlimited storage capacity: In cloud, users have unlimited storage capacity they can use when they need to.
- The environments of cloud computing are easily scalable.
- Efficient accidents response because backup recovery is very easy in IaaS Providers.

Limitations of Cloud Computing

- The language or Platform Constraints: Some SPs support specific languages or platforms only.
- Control of resources: The amount of resources that are controlled by the user differs greatly among providers depending on SPs polices.
- Security: Data security and privacy are the main concern, users don't have the control or don't know where their data is stored.
- Internet reliability: As we know, the provision of services in the cloud is over the Internet, when the Internet service is interrupted or when the connection is slow, this will impede access to these services. When these services are important work this will be a large problem, and the improvement of the Internet connection will reduce this problem. Also, we should be remembered that there is no guarantee of uninterrupted service.
- Dependence on the provider: A customer who uses cloud computing depends day after day on the SP to access information technology services rather than just maintenance and support. If the SP is in financial trouble, the ability to supply services may be affected.

- The maintenance and control are done by a third party. Thus, this reduces the confidentiality and security measures.
- Because the cloud is used with virtualization, an attacker may be able to access all data resources if he/she succeeds in attacking the hypervisor.

1.2 Data Security and Privacy

Security is the combination of confidentiality, the prevention of the unauthorized disclosure of information, integrity, the prevention of the unauthorized amendment or deletion of information, and availability, the prevention of unauthorized withholding of information [6]. This complements the CIA triad, a security model which provides a classic definition of security, in terms of the three key requirements for any secure system: confidentiality, integrity and availability hence the acronym CIA.

- *Confidentiality*: It is the ability to hide information from those people unauthorized to view it. It is the basis of many security mechanisms protecting not only information but other resources.
- *Integrity:* It is the ability to ensure that the data is accurate and the unchanged representation of the original information. This service protects data from malicious modification.
- *Availability*: It ensures that a resource is readily accessible to the authorized user upon the user's request

1.2.1 Security Issues in Cloud Computing

Security of the cloud is receiving much attention, as there have been many incidents of security and data breach. In 2018, a data breach on Facebook had the data of over 50 million users exposed after attackers exploited a vulnerability that allowed them access to users' personal data. British Airways also reported an incident of customer data theft between August and September 2018, ranging from personal information to financial details of customers making bookings and changes either on the website or the airline's app. Some of the security issues that need to be tended to before organizations switch completely to a cloud computing model as recognized by Gartner [7] are:

- *Data segregation:* Data in the cloud is in a shared environment alongside data from other customers. A system is thus required that separates data from different organizations and it should be provided by the CSP.
- *Long-term viability*: Ideally, your CSP will never go broke or get acquired and swallowed up by a larger company. But you must be sure your data will remain available even after such an event.
- *Recovery*: Each CSP ought to have a disaster recovery convention to ensure client data is protected in case of a disaster also and clients should ask about the mechanisms in place to achieve complete restoration and how long it would take.
- *Legal and Regulatory Compliance*: It may be difficult or unrealistic to use public clouds if your data is subject to legal restrictions or regulatory compliance. As best practices for cloud computing encompass a broader scope, this concern should disappear [4].
- *Insecure APIs*: The software Interfaces for the users to interact with the cloud services is also crucial to ensure the security of the cloud model. The API from the authentication and access control to the encryption and activity monitoring should be well implemented to protect against both accidental and malicious attack.

In a cloud environment, security and privacy are a cross-cutting concern for all cloud Actors, since both touch upon all layers of the cloud computing Reference Architecture and impact many parts of a cloud service. Therefore, the security management of the resources associated with cloud services is a critical aspect of cloud computing [4].

Cryptographic operations form one of the main tasks of secure management and are currently the most common techniques to achieve a satisfactory level of security requirements for cloud computing. Hence, while cloud services provide ubiquitous computing, elastic capabilities and self-configurable resources at lower costs, they also entail performing several cryptographic operations for the Secure Interaction of the consumer with various services provided and the Secure Storage of data generated/processed by those services. The key management system (KMS) required to support cryptographic operations for the above functions can be complex, due to differences in ownership and control of underlying infrastructures on which the KMS and the protected resources are located. In many instances, the KMS required for managing the cryptographic keys needed to protect that data have to be run on the computing resources provided

by the cloud Provider. This presents challenges to a cloud Consumer seeking to obtain the necessary security assurance from those cryptographic operations [4].

1.3 Cryptographic Protection of Data

The cloud storage providers claim that they supply the stored data with necessary security solutions. However, users often do not feel secure, because they have to trust the servers, while they do not exactly know what is going on inside the servers. Users lose their trusts even more, especially when they hear some news about, or become a victim of a security glitch. An example of such is the security breach that happened in Dropbox in June 2011. It was an authentication error that lasted several hours, which made it possible to access all users' data without a password [8].

The cryptographic mechanism must therefore be an approach that should make it possible for users to avoid thinking about server security. The only serious task the server has is to provide data availability. So in order to provide the needed security for the stored data in the cloud, we must ensure that the data is provided with cryptographic protection.

The following are the properties considered to achieve the desired system.

- Confidentiality: encryption of the data at the client side is preferred and should be performed just before storing or uploading the data to the cloud. The data also needs to be decrypted only after retrieving it from the cloud.
- Integrity: it would be more ideal to use digital signatures on the client side. This is preferred after encryption of the data. Verification of the signature has to be on the client side, after the data is retrieved from the cloud.
- Key exchange: a key exchange mechanism is required to enable access to data for cases where the data is shared among clients.
- All cryptographic keys are need to be just as secure as the data, except for the public key. This is because they are used to access the data. The key exchange mechanism therefore also needs to be as secured as the actual data.

- Access control: where we have the data being shared different levels of access to the data is granted. This will determine who has access to what keys that are used to access the data.

To provide solid security, and to have a secure file sharing mechanism, the above mentioned properties must be present in the cryptographic mechanism

1.4 Research Motivation

Although Cloud Computing provides a number of advantages, it also introduces a range of new security risks. Security of the cloud is receiving much attention, as there have been many incidents of security and data breach. In 2018, a data breach on Facebook had the data of over 50 million users exposed after attackers exploited a vulnerability that allowed them access to users' personal data. British Airways also reported an incident of customer data theft between August and September 2018, ranging from personal information to financial details of customers making bookings and changes either on the website or the airline's app. As Cloud Computing brings with it new deployment and associated adversarial models and vulnerabilities, it is clear that security becomes an area of great focus. This is especially true as Cloud Computing services are being used for e-commerce applications, medical record services, and back office business applications. To take full advantage of the power of Cloud Computing, it is anticipated that various security issues pertaining to the confidentiality, integrity and availability (CIA) of cloud computing have to be addressed. There have been several works done to address some of the issues of data security in the cloud, with focus of cryptography. There is however need to overcome some of the challenges that are still experienced and for more efficient techniques as compared to present schemes.

In light of the above statements, the following are pursued:

- To use Elliptic Curve Integrated Encryption Scheme (ECIES) to formulate a technique for ensuring Integrity of data stored in the cloud.
- To formulate a technique to ensure Confidentiality of data stored in the cloud using ECIES.
- To develop a scheme for efficient cryptographic key management using Identity Based Encryption (IBE).
- To develop and implement a data security scheme for Cloud Computing based on the hybrid ECIES and IBE.

1.5 Contributions of this Research

This thesis is written based on scientific papers, online sources, journals and Systematic Literature Review (SLR): Systematic Literature Review (SLR) is defined as identifying, evaluating and interpreting the available relevant work for a particular topic or phenomenon of interest. [9].

The scope of this thesis includes only technical Data security and privacy problems and solutions, excluding, among others legal and management aspects.

The main contributions of this thesis are:

- Describing Elliptic Curve Encryption (ECC) and how Elliptic Curve Integrated Encryption Scheme (ECIES) ensures integrity and confidentiality of cloud data.
- Describing Identity Based Encryption (IBE) and how it can be used to achieve better Key management in Cloud Computing.
- Presenting a hybrid data security scheme for cloud computing based ECIES and IBE.

1.6 Conclusion

In this chapter, an overview of cloud computing is given which includes the architecture, the essential characteristics of cloud computing, the types of clouds, the service and deployment models. The security issues in cloud computing are also presented, as well as the motivation and objective of the work.

Chapter 2. CRYPTOGRAPHY IN CLOUD COMPUTING

This chapter gives an introduction to cryptographic techniques used for ensuring security and privacy in Cloud Computing. This chapter also explores and presents the different resources available based on the SLR. It gives an overview of the fundamentals of cryptography, investigates the use of cryptographic mechanisms in cloud storage environments and briefly introduces the concepts of ECC and IBC.

2.1 Fundamentals of Cryptography

The word cryptography is derived from the Greek words *kryptos* which means "hidden", and *graphein*, which means "to write". Cryptography can thus be defined as the art of coding of information into hidden form or secret. It is the practice and study of techniques for secure communication in the presence of a third party known as an adversary- a malicious entity [10]. More generally, cryptography is about constructing and analyzing protocols that prevent third parties or the public from reading private messages [11]. For many years, cryptography was the exclusive domain of military, diplomatic and governmental secret services, and has been used to mainly provide security properties, such as data confidentiality, data integrity and data authentication. Cryptography has expanded due to the proliferation of computers and networks and the appearance of new cryptographic systems. In 1976, Whitfield Diffie and Martin Hellman presented the first asymmetric cryptographic algorithm, the Diffie-Hellman algorithm [12]. In 1978, Rivest, Shamir and Adelman defined their well-known RSA algorithm. Shamir continued publishing revolutionizing ideas, namely, threshold schemes, ID-based cryptographic systems and privacy homomorphism. Consequently, Koblitz and Miller have also independently proposed novel cryptographic schemes based on elliptic curve structures [12]. The sections that follow discuss the concepts of symmetric cryptography and public key cryptography.

2.1.1 Symmetric Cryptography

The pair of keys involved in message encryption and decryption algorithms is what gives the distinction between cryptographic schemes. Symmetric cryptography, also known as conventional cryptography, relies on the share of a secret key between two communicating entities Alice and Bob. This means the same key, the secret key, is used for both the encryption and decryption.

Let ***C*** be the cipher text message space, ***M*** the plaintext message space and ***K*** the key space.

The encryption algorithm, denoted by ***E,*** takes as input the plaintext message ***M***, and the secret key ***K***, and returns the cipher text ***C*** as shown below.

$$E: M * K \rightarrow C$$

The decryption algorithm, denoted by ***D,*** takes as input the cipher text message ***C***, and secret key ***K***, and returns the original message ***M.***

$$D: C * K \rightarrow M$$

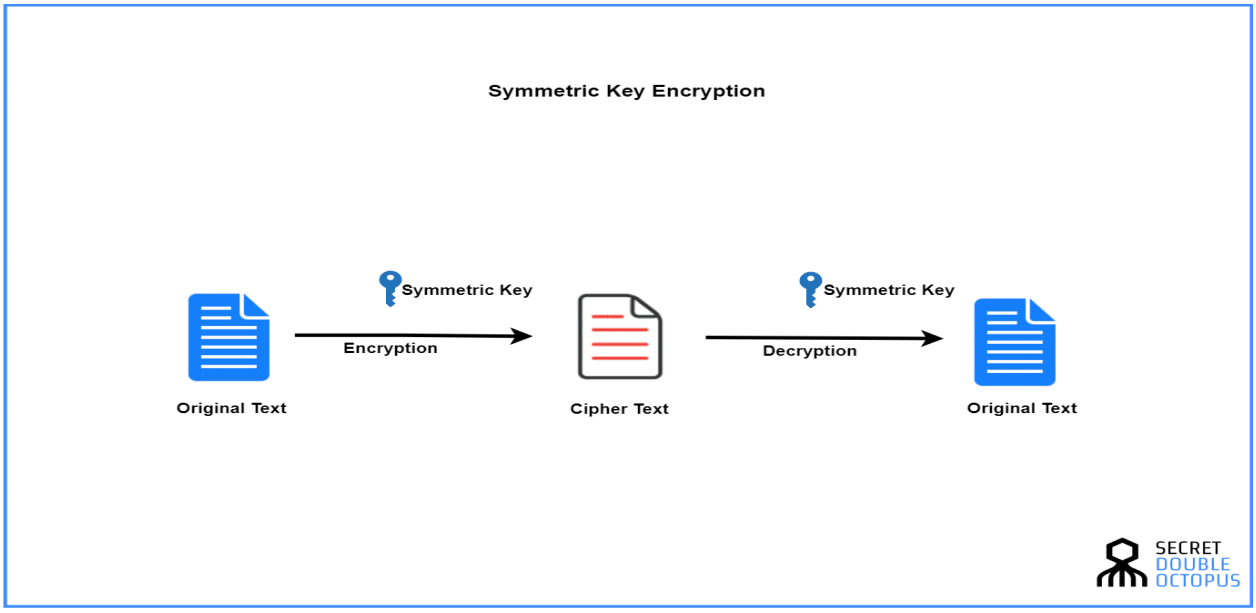


Figure 2.1 Symmetric Key Encryption [13]

There are two types of symmetric encryption algorithms grouped in terms of the cipher. These are;

- **Block ciphers.** Set lengths of bits are encrypted in blocks of electronic data with the use of a specific secret key. Blocks are commonly composed of 64 bits but can be larger or smaller depending on the particular algorithm being used and the various modes in which the algorithm might be capable of operating. As the data is being encrypted, the system holds the data in its memory as it waits for complete blocks. [14]

- **Stream ciphers.** Data is encrypted as it streams, one bit at a time, and are not retained in the memory of the system.

Majority of the encryption algorithms in use currently are block ciphers. Although block ciphers are often slower than stream ciphers, they tend to be more efficient. Since block ciphers operate on larger blocks of the message at a time, they do tend to be more resource intensive and are more complex to implement in hardware or software. Block ciphers are also more sensitive to errors in the encryption process as they are working with more data. An error in the encryption process of a block cipher may render unusable a larger segment of data than what we would find in a stream cipher, as the stream cipher would only be working with 1 particular bit. In general, several block modes can be used with an algorithm based on a block cipher to detect and compensate for such errors [14]. Typically, block ciphers are better for use in situations where the size of the message is fixed or known in advance, such as when encrypting a file or have message sizes that are reported in protocol headers. Stream ciphers are often better for use in situations involving data of an unknown size or the data is in a continuous stream [15]. Some of the popular symmetric encryption algorithms are:

Data Encryption Standard (DES)

DES was the first standardized cipher for securing electronic communications, and is used in variations (e.g. 2-key or 3-key 3DES) [16]. The original DES is not used anymore as it is considered too “weak”, due to the processing power of modern computers. Even 3DES is not recommended by NIST and PCI DSS 3.2, just like all 64-bit ciphers. However, triple DES (3DES) is still widely used in Europay MasterCard and Visa, EMV chip cards [16].

Advanced Encryption Standard (AES)

The most commonly used symmetric algorithm is AES, which was originally known as Rijndael. This is the standard set by the U.S. National Institute of Standards and Technology (NIST), in 2001 for the encryption of electronic data announced in U.S. FIPS PUB 197. This standard supersedes DES, which had been in use since 1977. AES uses three different ciphers: one with a 128-bit key, one with a 192-bit key, and the other one with a 256-bit key, all having a block length of 128 bits [14]. A variety of attacks have been attempted against AES, most of them against encryption using the 128-bit key. Most of them have been unsuccessful, and some partially successful. It is used by a variety of organizations, and is the replacement for DES as the standard encryption algorithm for the US federal government [15].

Symmetric encryption algorithms presume that Alice and Bob are able to exchange the key in a secure manner prior to each communication. As such, key management is a significant challenge in a multi-tenant environment, especially where we are having Security as a Service model (SecaaS) in cloud computing. Symmetric cryptography schemes are therefore usually mixed with public key algorithms which offer better key management.

2.1.2 Public Key Cryptography

In Public Key Cryptography (PKC), there is no longer a single key shared by the parties that are involved. Instead, the key generation algorithm produces multiple keys that are associated with another: usually, there is the secret key that should be known only by a single entity, and a public key that may be made known to everyone else.

The purpose of public key cryptography can be defined by the following characteristics:

- Confidentiality: only authorized users should be able to access the information to be protected.
- Integrity: ability to verify that a message has not been altered during transmission from the sender to the receiver.

- Authentication: the recipient of a message must be identified without any doubt by the message sender.
- Non-repudiation: an issuer of a message should not be able to deny authorship of the message.

In a PKC scheme, Bob, has a pair of keys: a public key and a secret private-key. To send messages to Bob, Alice and other users use Bob's public-key. Only Bob can recover the messages from the cipher texts by using his private-key.

This pair of keys is defined over a mathematical relation. The security of a PKC scheme is based on the assumption that it is computationally in-feasible to compute the private-key from the public-key. Such security assumption is assured by computationally hard mathematical problems such as the integer factorization problem, the discrete logarithm problem, the elliptic-curve discrete logarithm problem, and other several hard mathematical problems. Since 1976, when Diffie and Hellman introduced the concept of public key cryptography, there has been greater revolution in this branch of cryptography, which has led to the emergence of both cryptographic algorithms and cryptanalysis techniques of increasing complexity.

PKC schemes can be hard to implement due to very high computational requirements. Hence, in practice, cryptographic systems are often a mixture of symmetric key and PKC schemes. A public-key algorithm is always chosen for key establishment and then a symmetric-key algorithm is chosen to encrypt the communication data, thus achieving high throughput rates.

In general, one can divide practical public-key algorithms into three families:

- Algorithms based on the Integer Factorization Problem (IFP): given a positive integer n, find its prime factorization by expressing *n* in the form:

  $$n = P_1^{m1} P_2^{m2} \cdots P_k^{mk}$$

  Where the values pi represent prime numbers each with multiplicity order $m_{1 \geq 1}$

  Most popular example is RSA [17].

- Algorithms based on the Discrete Logarithm Problem (DLP): given α and β find positive integer *k* such that: $\beta = \alpha^{k} \pmod{p}$.

Examples are the Diffie-Hellman (DH) key exchange protocol and the Digital Signature Algorithm (DSA) [18].

- Algorithms based on Elliptic Curve Discrete Logarithm Problem (ECDLP): given points P and Q on an elliptic curve defined over a finite field, find positive integer k such that: $Q = k \cdot P$.

  Examples are the Elliptic Curve Diffie-Hellman (ECDH) key exchange protocol and the Elliptic Curve Digital Signature Algorithm (ECDSA) [18].

The computationally most intensive operation for RSA and Discrete Logarithm (DL) based PKC schemes are based on modular exponentiation, that is, the operation $x^e \bmod n$. These operations have to be performed using very long operands, typically 1024–2048 bits in length. However, for ECDLP systems, the operands are in the range of 160–256 bits in length. Table 2.1 below gives a comparison of the key sizes in bits, of RSA and ECC schemes.

ECC	RSA	Remarks
128 bits	700	Only short term security (breakable with some effort)
160 bits	1024	Medium term security
256 bits	2048	Long term security

Table 2.1 A comparison of key sizes needed to achieve equivalent level of security

Public-key encryption techniques may be used to establish a key for a symmetric-key system being used by communicating entities A and B. In this scenario, A and B can take advantage of the long term nature of the public/private keys of the public-key scheme and the performance efficiencies of the symmetric-key scheme. Since data encryption is frequently the most time consuming part of the encryption process, the public-key scheme for key establishment is a small fraction of the total encryption process between A and B. To date, the computational performance of public-key encryption is inferior to that of symmetric-key encryption. There is, however, no proof that this must be the case.

The important points in practice are:

- Public-key cryptography facilitates efficient signatures (particularly non-repudiation) and key management.
- Symmetric-key cryptography is efficient for encryption and some data integrity applications [17].

2.1.3 Public Key Infrastructure

PKI can be defined as a set of hardware, software, encryption technologies, people, procedures and policies required for managing, storing, creating, distributing, using, and revoking digital certificates and keys in cryptography. PKI is used for management of encryption keys and identities users. PKI merges public key cryptography, digital certificates, and certification authorities. The objective of a PKI is to manage certificates and keys [19]. The following are the functional components of a public key infrastructure:

- **Certification Authority (CA):** The CA acts as a trusted third party that issues digital certificates to be used by other parties. The responsibility of CA is to validate the identity of the person requesting the issuance of the certificate, and make sure that the certificate contains correct information and is digitally signed. The information contained in the digital certificates is issued by CAs like the subscriber name, the subscriber private and public keys, and the public key issued by CAs. This information depends on the company policy that issues the certificates. The CA validates the request for the certificate with the Registration Authority (RA) before it issued the digital certificate. The CA uses its own procedures in order to verify the validity of the certificate requests. These procedures depend on the policy of the organization and infrastructure available to validate the request. If the request is validated then the CA will issue the certificate [20].
- **Certificate Revocation:** The identity of users is authenticated by using certificates. All certificates have a validity period. Validity of a certificate refers to the time from when the certificate is issued until the time it expires [20]. Initially the application software must

make sure that the certificate is still trustworthy in the time of use. The CA must revoke the certificates which are no longer trustworthy. One of the reasons that may lead to the revocation of a certificate before the end of its validity period, could be that the private key corresponding to the public key in the certificate may be suspicious. Instead, an organization's security policy may dictate that the certificates of employees leaving the organization must be revoked. As a result, the PKI must include a scalable certificate revocation system [19].

- **Registration Authority (RA):** CA may use a third party -Registration Authority (RA) to perform the registration of users and accepting their requests for certificates. User registration is the process of collecting user information and verifying user identity, which is then used to register a user according to a policy [21].
- **Certificate Repository:** Certificate Repository is an internal database that stores all issued or revoked certificates, private Certificate Revocation List (CRL), pending certificate requests, etc. Only the CA or the RA can update this database. The major function of a Certificate Repository is to provide data which allows users to check the status of digital certificates for users who receive digitally signed messages [20] [22].

2.1.4 Digital Signatures

Digital signatures are used to ensure integrity of data, and they have the same principle as the handwritten signature. The difference is that if a digital signature is implemented properly, it is more difficult to fabricate than the handwritten signature. To apply a digital signature to messages, public key encryption is used. For instance, Alice may want to send a message to Bob. The message can be encrypted or not, but people usually encrypt the message to ensure secrecy. Then she generates a pair of keys, that is a private key and a public key. She keeps the private key secret, and publishes the public key. She signs her message using Bob's public key, and then she sends the signed message to Bob. When Bob receives the message, he tries to verify the signature by using his private key. If the signature is verified successfully, then he is sure that the message is untouched and the actual sender of the message is Alice. If the verification is failed, then Bob knows that either the message has been tampered with, or it is not sent by Alice at all. [23]

In practice, usually the message itself is not signed. A hash function is used to hash the message, and by this, a short digest is produced, which is then signed and attached to the message as a signature.

For the process of signing and verifying we need three algorithms:

- A key generation algorithm: given a security parameter, this generates a private key and a public key.
- A signing algorithm: takes the data and a private key, and outputs a signature.
- A verifying algorithm: takes the data, a public key and a signature, and it outputs either success or failure for the verification.

We also need another algorithm, hash function algorithm, in order to generate the hash code.

The most well-known signature schemes that are used with regard to digital signature are RSA signature scheme, DSA, and Elliptic Curve Digital Signature Algorithm (ECDSA). All these three signature schemes contain the necessary algorithms mentioned above.

2.2 Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) is a cryptographic scheme that uses the properties of elliptic curves to generate cryptographic algorithms. This was proposed in 1985 by Koblitz and Miller using the group points on an elliptic curve defined over a finite field in discrete logarithmic cryptosystems [12]. ECC gives same level of security as that of RSA and ElGamal cryptosystems but with smaller key size. This makes it a better and more efficient solution as compared to previously deployed techniques.

An elliptic curve is the solution set over a non-singular cubic polynomial equation with two unknowns over a field F. In other terms, an elliptic curve E is the graph of points on the plane curve defined by the *Weierstrass equation*:

$$y^2 = x^3 + Ax + B$$

More and in-depth description and workings of elliptic curves and the arithmetic of elliptic curves and their use in cryptography is presented in the next chapter.

2.3 Identity Based Cryptography

In 1984, ID-Based Cryptography (IBC) was introduced by Shamir [24] with the original idea to provide public and private key pairs with no need for certificates and Certificate Authority (CA) deployment. The IBE scheme is a public-key cryptosystem where any arbitrary string is a valid public key. The corresponding private keys must be computed by a trusted third party called the private key generator (PKG), who possesses a master secret. Users of the system therefore request their private key from the PKG.

It is however undesirable for a large network like the cloud, to have a single PKG, because then it means the PKG has a burdensome job. Not only is private key generation computationally expensive, but also the PKG must verify proofs of identity and must establish secure channels to transmit private keys [25]. Hence the need for Identity-based encryption (IBE).

In Identity-based encryption systems approach, the trusted third party called Private Key Generator (PKG) will create a master private key and its corresponding master public key. The PKG will then publish the master public key and keep the master private key. Any user can then generate his public key by combining the master public key and their identity value. The user then connects the PKG with their identity to obtain their private key. The PKG uses the master private key and user's identity to create user's private key [26]. Franklin, Boneh, and Cocks defined four types of algorithms for IBC system. These algorithms contain setup, extract, encryption and decryption [27].

Figure (2.2) below shows the Identity Based Encryption scheme.

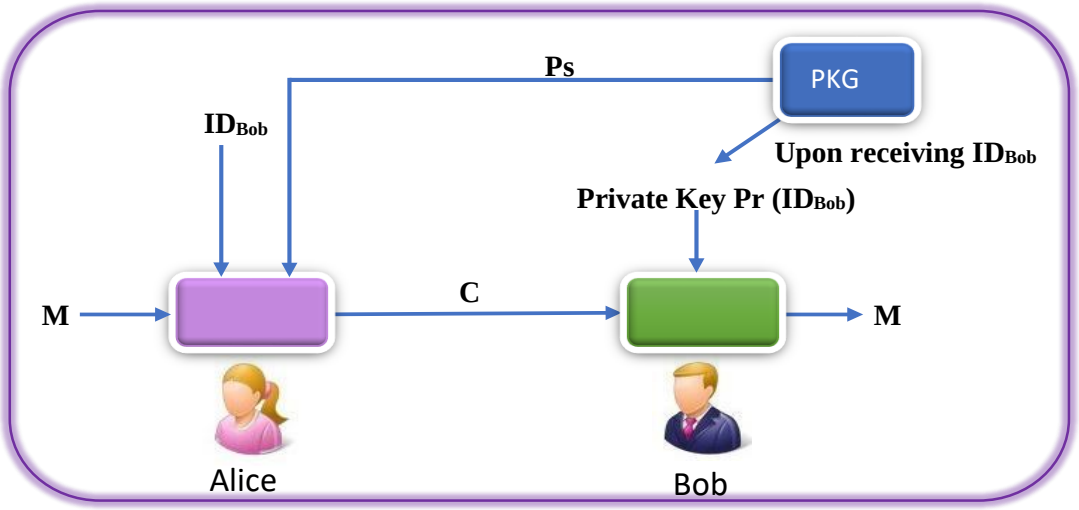


Figure 2.2 Identity Based Encryption scheme

- **Setup:** In this algorithm a PKG will generate a master private key (Pr) as well as the parameters of system Ps. Pr must be kept secret to be used to create the user's private key. Ps is made public to all users and is used to create a public key for users with their identities [28].

- **Private Key Extraction:** When Bob requests his private key from the PKG, PKG will use the identity of Bob (ID_{Bob}), system parameters Ps and master key Pr to create Bob's private key [28].

- **Encryption:** When Bob wants to encrypt a message and send it to Alice, he can use the system parameters Ps, Alice's identity and the message as input to generate the cipher text
- **Decryption:** Upon receiving the cipher text C from Bob, Alice decrypts it using her private key Pr to recover the plaintext M [28].

In identity based signature (IBS) systems, when Alice wants to sign a message, at first she gets the private key which is associated with her identifier from the PKG. Then she signs a message by using this private key. Bob will verify from Alice's signature by using her identifier information [28]. Figure (2.3) describes IBS scheme.

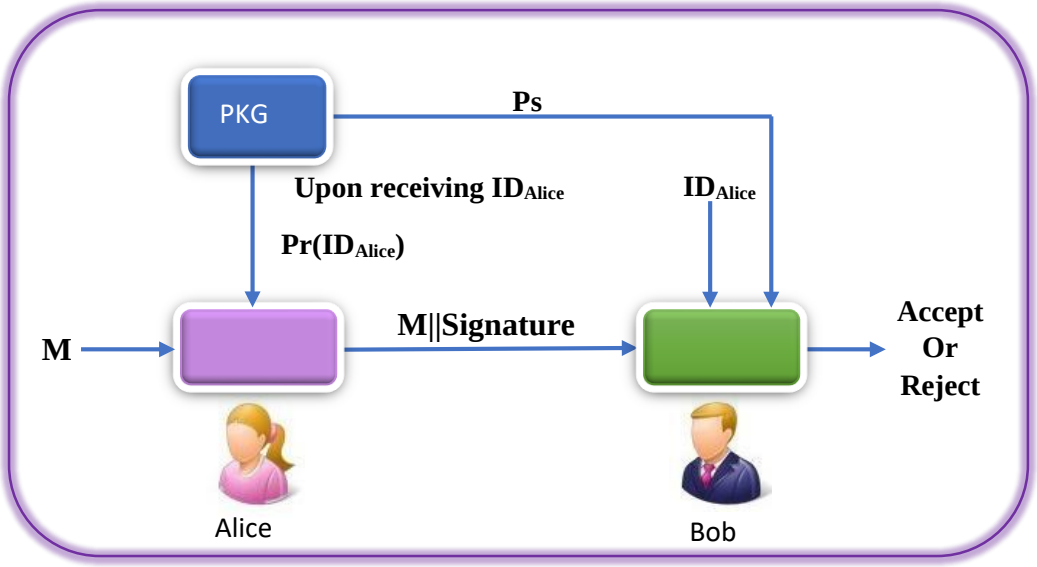


Figure 2.3 Identity Based Signature scheme

IBC also consists of four types of algorithms. These are; setup, private key extraction, signature generation and signature verification [28].

- **Setup:** PKG will generate a master private key (Pr) as well as the system parameters Ps.
- **Private Key Extraction:** Alice gives her ID to PKG. The PKG will use her identity (ID_{Alice}), Ps and Pr to produce Alice's private key $Pr(ID_{Alice})$.

- **Signature Generation:** When Alice wants to sign her message M, she uses her private key $Pr(ID_{Alice})$ to create her signature S.
- **Signature Verification:** After Bob gets a signed message from Alice he checks the validity of the signature S using ID_{Alice} and system parameters Ps. if the signature S is true, then he returns Accept. Otherwise, he returns Reject.

Identity Based Cryptography is considered the next generation of public key cryptography because it completely eliminates the need to use of certificates. The first and the clearest advantage of this security scheme is that the users can communicate securely by encrypting and signing the messages without the need to exchange public keys through the exchange of certificate. This way, there is no longer need for a certificate distribution infrastructure, which is the PKI. Another advantage is that the certificate validation step is also missing because there is no need to check if the public key actually belongs to the user by checking the signature of the certification authority. The keys also expire, and therefore do not need to be revoked. In a traditional public-key system, keys must be revoked if compromised [29].

There are however some disadvantages that come with IBC schemes. The user receives their private key from the PKG which is computed as a function of the master secret and the user's identity. This requires a user to authenticate themselves to the PKG. This also presents the need for a secure channel to be used by the PKG to send the user their private key. The user's PKG must publish parameters that embed its master secret key, and the message recipient, Alice, must obtain these parameters before sending an encrypted message to Bob. Key escrow is the other disadvantage of IBC scheme. This is due to the fact that the PKG knows or can be able to compute the private keys of all the users. Users are therefore forced to blindly just trust the PKG with their keys.

The proposed system eliminates the need for certificates. Since the keys expire after the end of the session, there is no need for an algorithm for key revocation. The use of a trusted authority (PKG) to generate the parameters of the system and master secret key reduces the complexity.

Since the users generate their own keys with the help of a trusted authority, the proposed system overcomes the problem of key escrow.

2.4 Related Works

Several works related to the aims of this thesis, present data security in cloud computing as follows: In 2011, Suli Wang et al. proposed a method for file encryption and decryption based on RSA algorithm with smaller sizes [30]. Syam Kumar and Subramanian proposed an effective and safe method by using ECC and Sobol sequence in 2011. This method allows third party auditors to periodically verify the data integrity stored at CSP without retrieving original data [31].

Arjun Kumar et al. [32] proposed a method that allows user to store and access the data securely from the cloud storage by use ECC. This proposal divided the storage to two section the first section to store user private data, and the second to store shared data and use pin number with secret key to encryption data.

In 2012, Abbas Amini proposed system for securing storage in cloud computing. This proposal uses RSA algorithm for data integrity, and uses AES algorithm to achieve confidentiality of the stored data [33]. K. Govinda and E. Sathiyamoorthy proposed a manner of securing data storage and ensuring identity anonymity in a private cloud by using GDS (Group Digital Signature). They used the concept of key exchange with Diffie-Hellman protocol and strong RSA algorithm for the keys generation in addition to the process of signature, encryption and decryption [34].

In 2014, Puneetha C and M Dakshayini proposed data security model using ECC algorithm and hash function as digital signature [35]. Swarnalata Bollavarapu and Bharat Gupta [36], proposed a data storage security system in cloud computing that uses RSA, ECC and RC4 for encryption and decryption techniques.

A framework for double authentication techniques and specialized procedures that can effectively protect the data during the period of transition from the user to the cloud was also proposed by Nagendra Kumar, Ashok Verma and Ajay Lala in 2014. This utilized DS (Digital Signature) using RSA algorithm [37].

Noha MM. Abdelnabi et al. [38], used a hybrid of cryptographic algorithms. This combination contains RSA and AES algorithm to provide confidentiality and authentication, and used SHA256 to generate a signature. The proposed scheme provided the three security primitives – authentication, confidentiality and integrity.

Divya Prathana Timothy and Ajit Kumar Santra [39], designed a new cryptographic scheme for data security in cloud based on a hybrid cryptosystem. This cryptosystem was comprised of secret key and public key cryptographic algorithms. In this scheme the Blowfish algorithm was used for data security while RSA algorithm was used for user an authentication. Secure Hash-2 function was also used in this scheme to provide data integrity.

Zinah Raad Saeed et al [40], proposed a scheme that used a combination of secret key and public key models, consisting of AES, RSA and ECC cryptographic methods. This system provides high level of security for cloud computing in terms of encryption and authentication. In this study, the data is encrypted using AES. The keys are then encrypted using ECC, after which RSA is used to encrypt both the ciphertext and the encoding key again.

.

Chapter 3. METHODOLOGY AND DESIGN

Cryptographic mechanisms based on elliptic curves depend on arithmetic involving the points of the curve. In this chapter, the idea of elliptic curves cryptography and the basic arithmetic of elliptic curves over arbitrary algebraically finite fields is covered. In the first section, the definition of an elliptic curve is given together with the group law on such curves. An explicit algorithm for the group law of these curves is also presented. All this is to show proof and reason for the choice of ECC as the methodology used for this research thesis.

3.1 Elliptic Curves Arithmetic

3.1.1 The Weierstrass equation

Elliptic curves are curves of genus one, that have a specified base point and are defined by the *Weierstrass equation* which is an equation of the form:

$$y^2 = x^3 + Ax + B$$

The elements A, B, x, and y are usually taken to be elements of a field, for example, the real numbers **R**, the complex numbers **C**, the rational numbers **Q**, one of the finite fields $\mathbf{F}_p$ for a prime p, or one of the finite fields $\mathbf{F}_q$, where $q = p^k$ with $k \geq 1$. If K is a field with $A, B \in K$, then we say that E ***is defined over*** K. Throughout this thesis, E and K will implicitly be assumed to denote an elliptic curve and a field over which E is defined [41].

It is clear that the projective point $\boldsymbol{O} := (0 : 1 : 0)$ is the only point at infinity on an elliptic curve. We call this point the point at infinity. Elliptic curves are therefore handled as affine curves, and the point at infinity is treated separately.

3.1.2 Discriminant

The discriminant, denoted by Δ, of a Weierstrass equation is the quantity given by

$$\Delta = -16(4A^3+27B^2)$$

This is used to determine whether an arbitrary Weierstrass equation defines an elliptic curve and this is if and only if $\Delta \neq 0$. This is used to prove the non-singularity of elliptic curves to ensure three distinct roots of the curve. Only nonsingular curves are cryptographically useful, as singular curves provide no extra benefit over finite fields, and have slower operations.

3.1.3 Group Laws

The use of elliptic curves in cryptography depends on the fact that the points on an elliptic curve meet the conditions of an Abelian group structure, and this allows for efficient computation. The group operations that fulfill the group laws of an Abelian group are:

- Point addition
- Point doubling
- Scalar multiplication

We define Point addition on an elliptic curve, *E*, defined over a field *K* given two points, $P_1 = (x_1,y_1)$ and $P_2 = (x_2,y_2)$ on the elliptic curve *E*. We then find a third point, P_3 by drawing a line L between the points P1 and P2. The point of intersection, which is the third point on *E*, denoted as P'_3. We then reflect P'_3 along the x-axis to get P_3. This operation for point addition is expressed as:

$P_1 + P_2 = P_3$

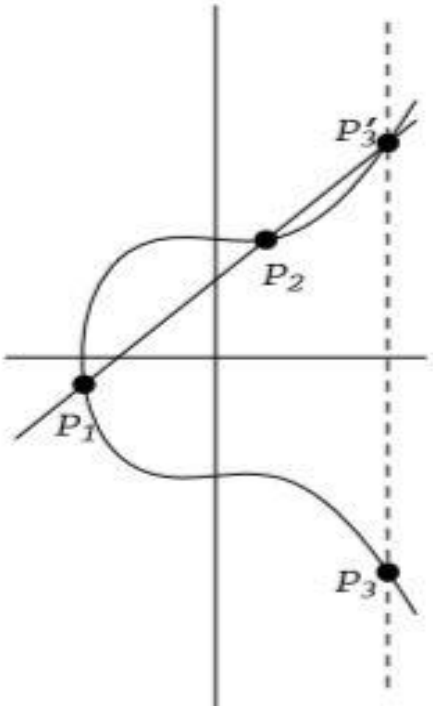


Figure 3.1 Points addition on an elliptic curve [42]

Given $x_1 \neq x_2$, then $x_3 = m^2 - x_1 - x_2$, $y_3 = m(x_1 - x_3) - y_1$

Where m (Slope of the line L) = $(y_2 - y_1) / (x_2 - x_1)$

If $x_1 = x_2$ but $y_1 \neq y_2$, the line through P_1 and P_2 is a vertical line, which therefore intersects E in ∞. Reflecting ∞ across the x-axis yields the same point∞. Therefore in this case, $P_1 + P_2 = \infty$.

If $P_1 = P_2$ and $y_1 \neq 0$, then

$$x_3 = m^2 - 2x_1, \quad y_3 = m(x_1 - x_3) - y_1, \text{ where } m = (3x_1^2 + A) / 2y_1$$

This is where we have Point doubling.

If $P_1 = P_2$ and $y_1 = 0$, then this shows that the line L is a vertical line and we therefore have $P_1 + P_2 = \infty$.

Also $P + \infty = P$, for all points P on E.

The right hand side of the Weierstrass equation is cubic. This is the reason why the line between any two points will intersect at a third point, the first step in the operation. Then the y^2 term on the left hand side makes the curve symmetric about the *x-axis*, which is vital for the reflection part [41].

The addition of points on an elliptic curve E, satisfies the following properties:

- Commutativity: $P1 + P2 = P2 + P1$ for all $P1, P2$ on E.
- Existence of identity: $P + \infty = P$ for all points P on E.
- Existence of inverses: Given P on E, there exists P' on E with $P + P' = \infty$.This point P' will usually be denoted by $-P$.
- Associativity: $(P1 + P2) + P3 = P1 + (P2 + P3)$ for all $P1, P2, P3$ on E.

The points on E therefore form an additive Abelian group with ∞ as the identity element. [41]

3.1.4 Elliptic Curve Discrete Logarithm Problem (ECDLP)

The elliptic curve discrete logarithm problem (ECDLP) can be expressed as: given an elliptic curve E defined over a finite field F_q, a point $P \in E(F_q)$ of order n, and a point $Q \in (P)$, find the integer $l \in [0, n-1]$ such that:

$$Q = lP, \text{ where } lP = \underbrace{P + P + \ldots + P}_{l \text{ times}}$$

The integer l is called the discrete logarithm of Q to the base P, denoted by:

$$l = \log_p Q.$$

It is believed that the usual discrete logarithm problem (DLP) over the multiplicative group of a finite field and ECDLP are not equivalent problems, and that ECDLP is significantly more difficult than DLP. The main reason is that there is no known sub exponential-time algorithm to solve ECDLP in general.

3.2 Elliptic Curve Integrated Encryption Scheme (ECIES)

This is a public key encryption scheme, which was invented by Abdalla, Bellare, and Rogaway [11]. It has been standardized in ANSI X9.63 and ISO/IEC 15946-3, and is in the IEEE P1363a. This scheme provides safety against adaptive chosen-plaintext and chosen-ciphertext attacks. It provides capabilities for encryption, key exchange and digital signature together. It is therefore called Integrated Encryption Scheme, since it is a hybrid scheme that uses a public key system to transport a session key for use by a symmetric cipher [42].

In ECIES, a Diffie-Hellman shared secret is used for generating two symmetric keys, k1 and k2. The key k1 is used for encrypting the plaintext using a symmetric-key cipher, while the key k2 is used for authenticating the resulting ciphertext. ECIES uses the following cryptographic functions [43]:

- A Key Derivation Function (KDF) that is constructed from a hash function *H*. If a key of *x* bits is required then the KDF(S) is defined to be the concatenation of the hash values $H(S,i)$, where *i* is a counter that is incremented for each hash function evaluation until *x* bits of hash values have been generated. [26]
- ENC and DEC. ENC is the symmetric encryption algorithm function, while DEC is the decryption function.
- MAC: the Message Authentication Code algorithm used to authenticate messages

If Alice wants to send a message *m* to Bob, Bob will have to establish his own public key. He does so by choosing an elliptic curve *E* over a finite field $\mathbf{F_q}$. He then chooses a point *A* on the elliptic curve *E*, which should be of a large prime order *N*. Then he chooses a secret integer *s* and computes $B = sA$.

The public key is thus given by (q, E, N, A, B) while the private key is *s*.

ECIES is considered an enhancement over ElGamal public key method due to the advantage it has over ElGamal. This is because the message is not represented as a point on the curve.

3.3 The Proposed System

The proposed system aims to combine the security of IBC and ECC by using a Trusted Cloud (TC). IBC is used as it significantly lowers the complexity of key generation. This is also because when using IBC, we do not need a certificate to be issued. The use of TC also helps in combatting issues such as denial of service attacks (DoS) on the cloud, especially considering the TC has all of the users' information. All these parts increase the strength and resistance of the system. The private key generator (PKG) and the trusted cloud (TC) are considered as a Trusted Authority (TA) in this system.

3.3.1 Essential parts of the proposed system.

The proposed system consists of three parts: TA, user, and TC. CSP management, access control, and user authentication are done by Trusted Cloud, while the TA is used to help the user to generate users' private and public keys. The general architecture of the proposed system is shown in figure (3.2) below.

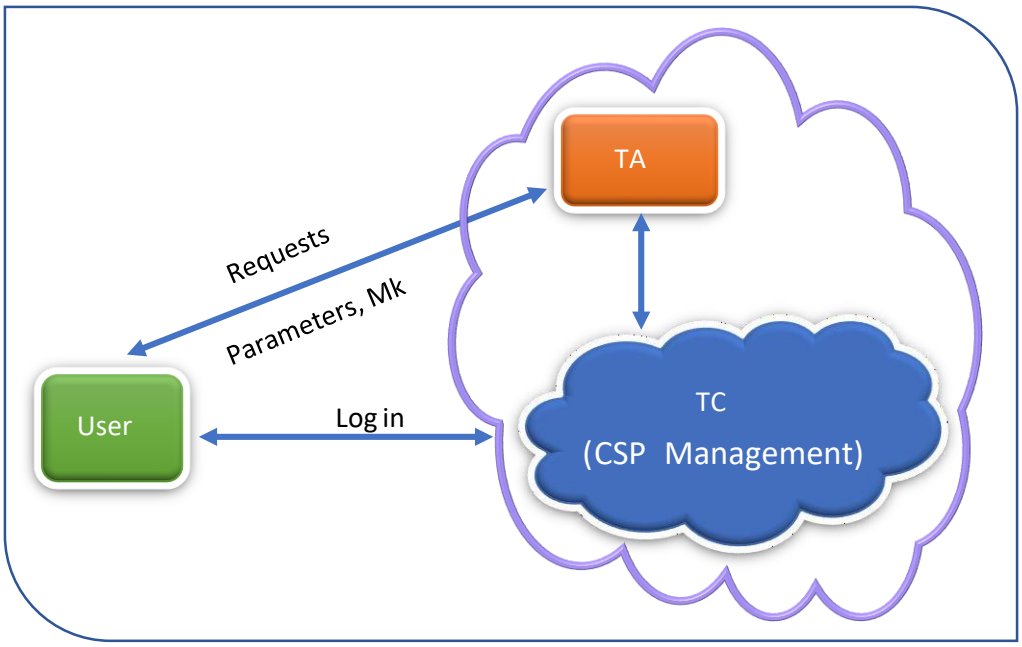


Figure 3.2 General structure of proposed system

The main purpose of this proposed system could be summarized by the following points:

- Providing more secure method to protect user's files by encrypting them using Elliptic Curve Integrated Encryption Scheme (ECIES) algorithm.
- Reducing the complexity by using IBC, and providing better security.
- The system should be strengthened by achieving the major three security goals that are integrity, confidentiality, and authentication.

3.3.1.1 Trusted Authority (TA)

The trusted authority is the most important part in this system. It is what generates the essential parameters in the system. The parameters are the base point (P), the field ($\mathbb{F}_p$), the prime number (p), the order (n), the curve (E), and the curve's parameters (a,b). Moreover it is also

used to generate a random private number we call master key (Mk) which it keeps secret. When the user wants to generate the public and private keys, they send their request to the TA. The TA will generate the system parameters and the master key (Mk), and then sends them back. Figure (3.3) shows the trusted authority operations.

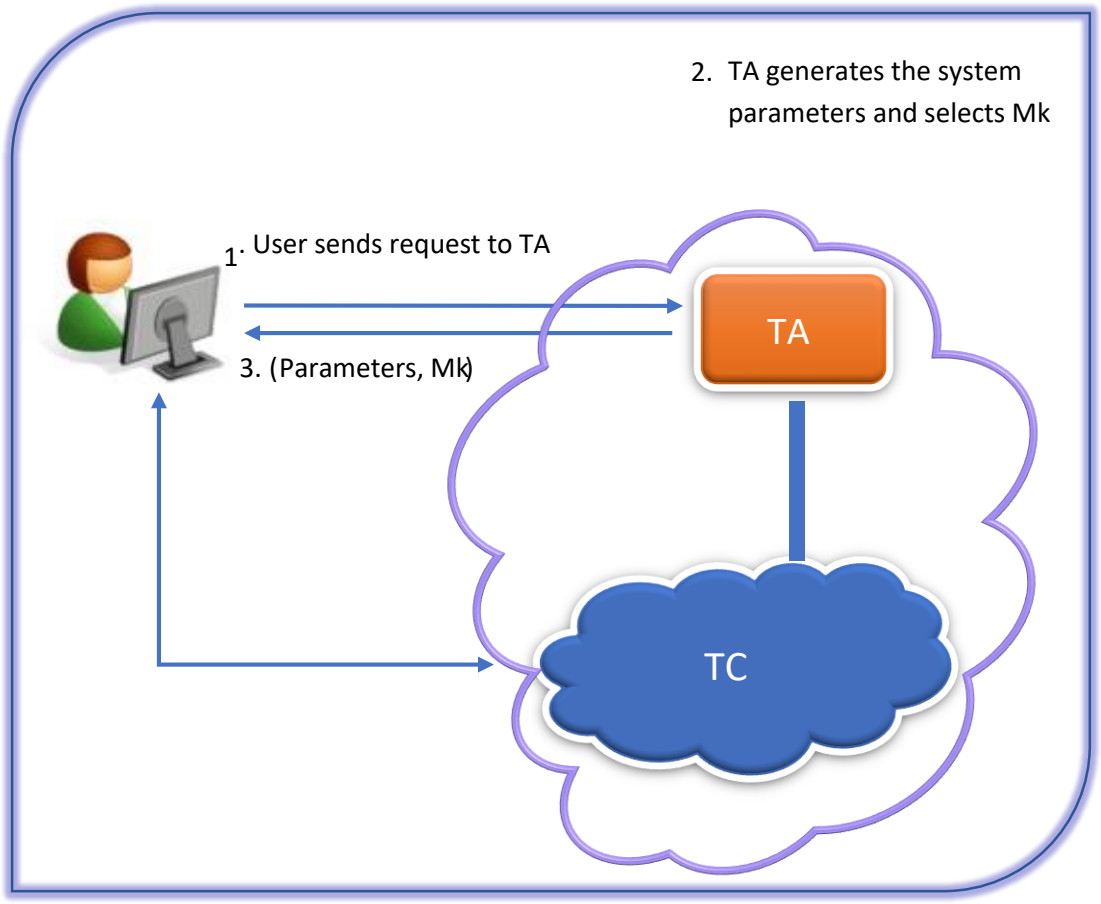


Figure 3.3 Trusted Authority operations

3.3.1.2 The Client

The second part of this system is the client (user), which is responsible for generating private and public keys. The first action performed by user is Login operation, the user sends their identity and password to TC. The TC matches user's information with its database, if the user

is not registered, the TC rejects them and shows the message (Not valid user name or password), but if they are registered, the TC allows them to enter the system. When the user enters the system they encrypt a file, and thus the need for encryption keys.

When the user needs to use any service from the system, they must at first generate cryptographic keys. They must acquire the parameters of system and Mk from the TA to be able to generate these keys. After which they compute the hash value (H) to the user's identity (ID_U) and get the private key (Pr_U). This is obtained from multiplying the Mk with the $H(ID_U)$. Then by the use of this private key (Pr_U), the user generates the public key (PU_U). This is done using the elliptic curve discrete logarithm problem.

When the user wants to encrypt a file, the user can encrypt the file locally by using ECIES algorithm. This algorithm consist of four algorithms (ECC, ECDH, AES, and MAC). At first it uses elliptic curve (ECC) to generate public and private key, then uses ECDH algorithm to generate a shared secret key (Sk). After that the key derivation function (KDF) will use Sk to derive two keys, the first key is the encryption key (k1) and the second is MAC key (k2). The k1 is then used to encrypt the file by using AES-128 to generate the ciphertext (C). This is because AES-128 is relatively more secure than AES-192 and AES-256 and is recommended for devices with limited resources [44]. After that, we compute the MAC value to this ciphertext by using k2 to produce the tag. Figure (3.4) shows file encryption operations.

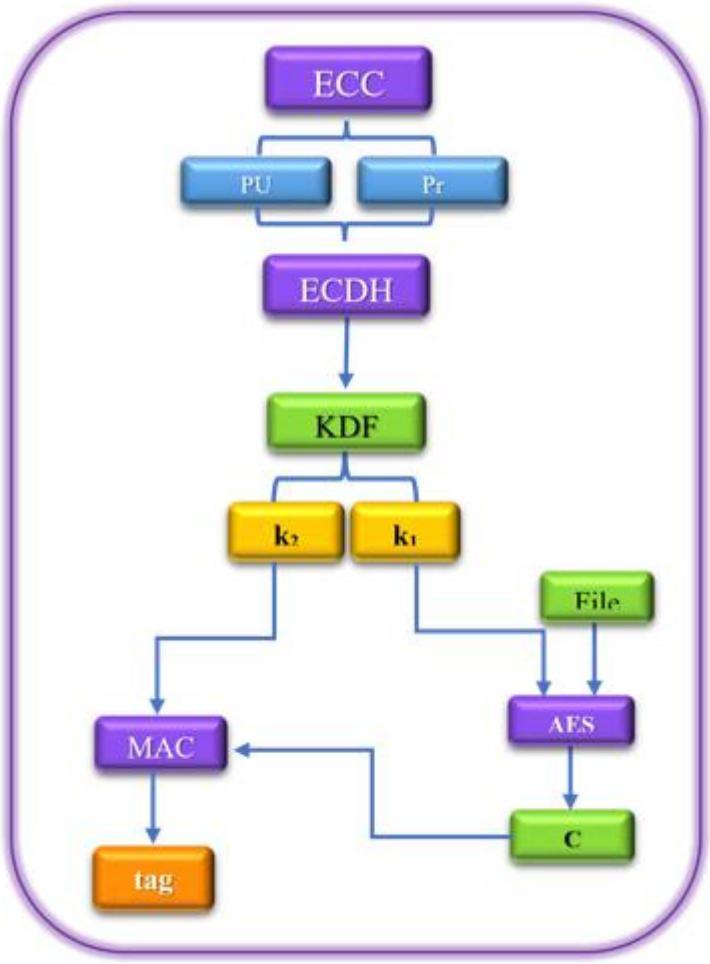


Figure 3.4 File Encryption

In decryption, the user will receive the sender's public key, C, and a tag. At first the user generates the public and private keys then uses ECDH to generate (Sk). The KDF then derives k1 and k2. After that, there is a comparison of the tag value with the result of MAC to ensure authentication. Then we have the decryption of the file using AES to produce the plaintext. Figure (3.5) explains the decryption operations.

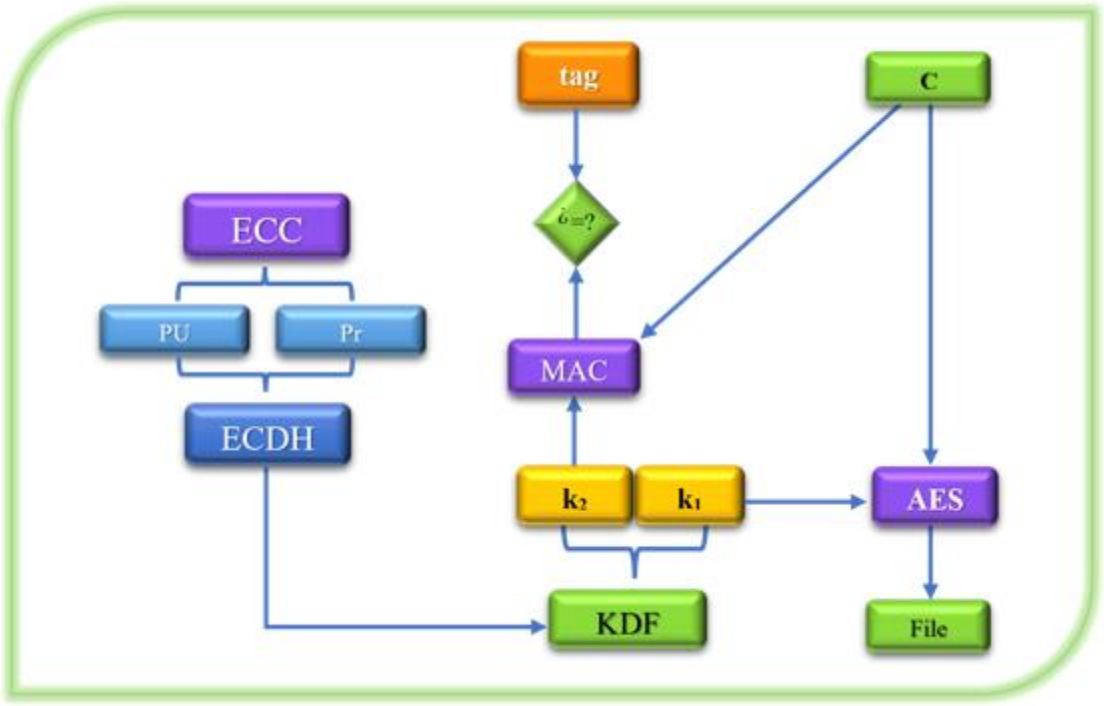


Figure 3.5 File Decryption

3.3.1.3 Trusted Cloud (TC)

The trusted cloud is the final part in the system. It is responsible for the CSP management processes. TC has a database containing all user information such as the user identity, password and users' email. When a user logs in, the TC is what verifies the user's identity and password by comparing it with the information in the database. The TC is also used to manage the CSPs, and enables updates. For example, if a CSP adds a new branch, the TC must register these changes in its database after authentication and trust from these CSPs.

Figure (3.6) explains the operations of TC.

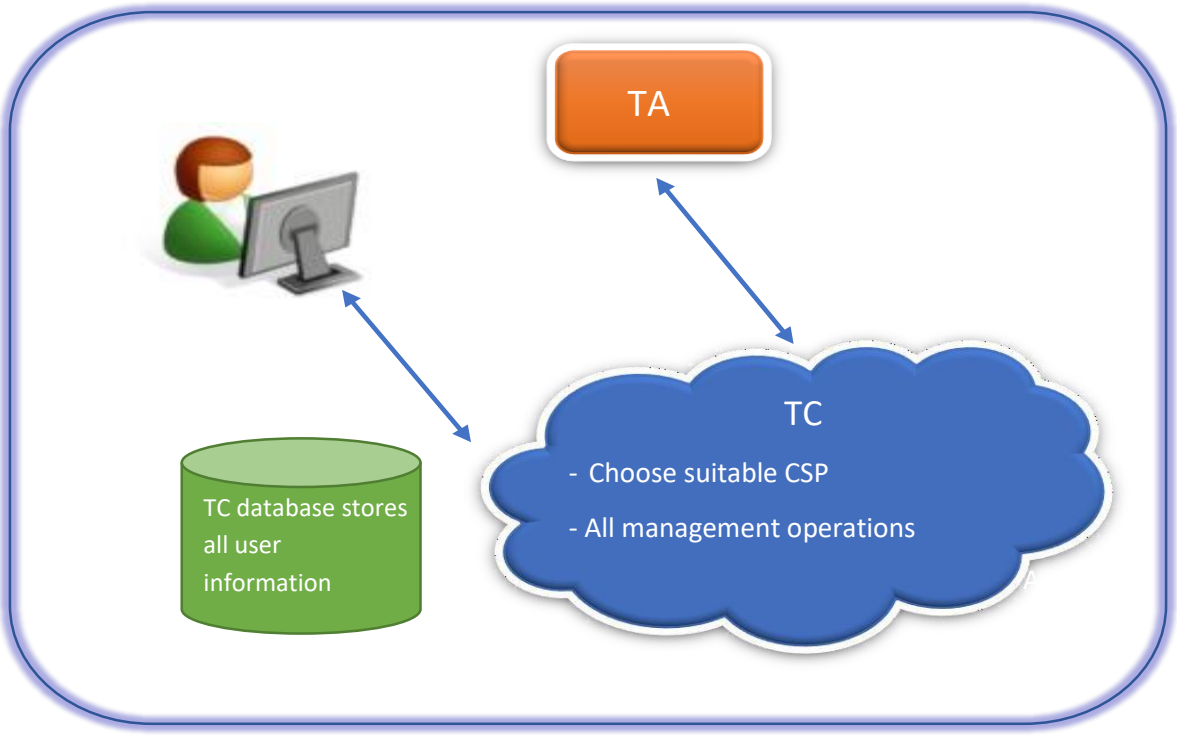


Figure 3.6 Operations of the Trusted Cloud

3.4 Proposed Cryptographic Schemes

This section describes how the security base is applied to the proposed system. The main goal of this security base is to prevent an adversary from intercepting users' data so as to retain the privacy between the users. The basic goal of proposed cryptographic scheme could be summarized by the following points:

1. Securing keys exchange.
2. Securing file encryption.

The proposed security base uses (ECC512, ECDH, ECIES, ECDSA, and SHA). The use of ECC provides confidentiality, ECIES provides confidentiality and authentication, ECDH provides secure key exchange, ECDSA provides data integrity, and SHA provides authentication.

To encrypt or decrypt any file the following steps are done:

At Encryption:

- The user generates k1 and k2.
- Encodes file's data (using ASCII code).
- He applies ECIES algorithm to these data.
- Saves the encrypted file.

At Decryption:

- The user generate the decryption keys.
- Verify from authentication.
- Decrypts the file and retrieves the original file.

3.5 The Proposed System Algorithms

This proposal implements and defines all required security protocol to make the file encryption system secure and robust against attack. Some of the basic cryptographic algorithms implemented by the system are:

3.5.1 Public and Private Key Generation Algorithm

To generate the public and private keys, the user needs to get the system parameters from the TA. The TA will generate the parameters (p, n, $\mathbb{F}p$, E, P, a, b) and select a random integer private number in the range 1- (p-1) to act as Mk. After that, the user receives (p, n, $\mathbb{F}_p$, E, P, a, b, Mk) from the TA, the user then computes the hash value to their identity (ID_U) and multiplies the result by Mk to generate the user's private key. The user then multiplies this private key by the base point to obtain their public key. This section explains an algorithm which is used to generate public and private keys.

Public and Private Key Generation

INPUT: Elliptic curve domain parameters (**p**, **E**, $\mathbb{F}_p$, **P**, **n**), $\mathbf{ID_U}$, **Mk**.

OUTPUT: Public key $\mathbf{PU_U}$ and private key $\mathbf{Pr_U}$.

Step 1: Setup: the user receive the parameters of system and Mk.

Step 2: Extract:

1. Compute the hash value of user identity (ID_U)

   $\mathbf{Q_U = H (ID_U)}$

2. Compute private key by multiply **Mk** with $\mathbf{Q_U}$

   $\mathbf{Pr_U = Mk*Q}$

3. Compute public key by multiplying the base point **P** with private key $\mathbf{Pr_U}$ $\mathbf{PU_U = Pr_U*P}$

3.5.2 File Encryption and Decryption Algorithms

When the user needs to encrypt a file, they generate the private and public keys after sending a request to the TA to get the parameters and Mk. The user then computes the shared secret key by using ECDH. After that, the KDF is used to generate the encryption and MAC keys, after which they encrypt the file. The following algorithms explains the operations that are used to encrypt and decrypt a file.

File Encryption
INPUT: the plaintext **m**.
OUTPUT: public key $\mathbf{PU_A}$, Ciphertext (**C)**, Tag (**t)**.
Step 1: Use algorithms 4.1 and 4.2 to generate public and private key.
Step 2: Compute shared secret key using ECDH $\mathbf{Sk = Pr_A * PU_B}$.
Step 3: Use the KDF to get a symmetric encryption key ($\mathbf{k_1}$) and a MAC key ($\mathbf{k_2}$) ($\mathbf{k_1,k_2}$)= $\mathbf{KDF\ (x_{Sk}, PU_A)}$, where $\mathbf{x_{Sk}}$ is the x-coordinate of **Sk**.
Step 4: Encrypt the plaintext **m** by using AES 256: $\mathbf{C = ENC k_1\ (m)}$.
Step 5: Compute the tag encrypted data: $\mathbf{t= MAC k_2\ (C)}$.
Step 6: Return $\mathbf{(PU_A||C||t)}$.

In the decryption side, the user generates the public and private keys by using ECC and then generates a shared secret key using ECDH. The KDF then extracts k1 and k2 to verify for authentication, and decrypt the encrypted file.

File Decryption

INPUT: (PU_A||C||t).

OUTPUT: the plaintext **m**.

Step 1: Use algorithms 3.1 to generate public and private key.

Step 2: Compute the shared secret key using ECDH **$Sk = Pr_B * PU_A$**.

Step 3: Use KDF to derive a symmetric encryption key (k_1) and MAC key (k_2)

(k_1,k_2)= **KDF (x_{Sk}, PU_A),** where x_{Sk} is the x-coordinate of **Sk**.

Step 4: Verify from the authentication of tag encrypted message: **$t' = MACk_2$ (C)**.

Step3: Decrypt the Ciphertext **$m = DECk_1$ (C)**.

Step4: Return the plaintext **m**.

The figure (3.7) below shows the overall operation of the system.

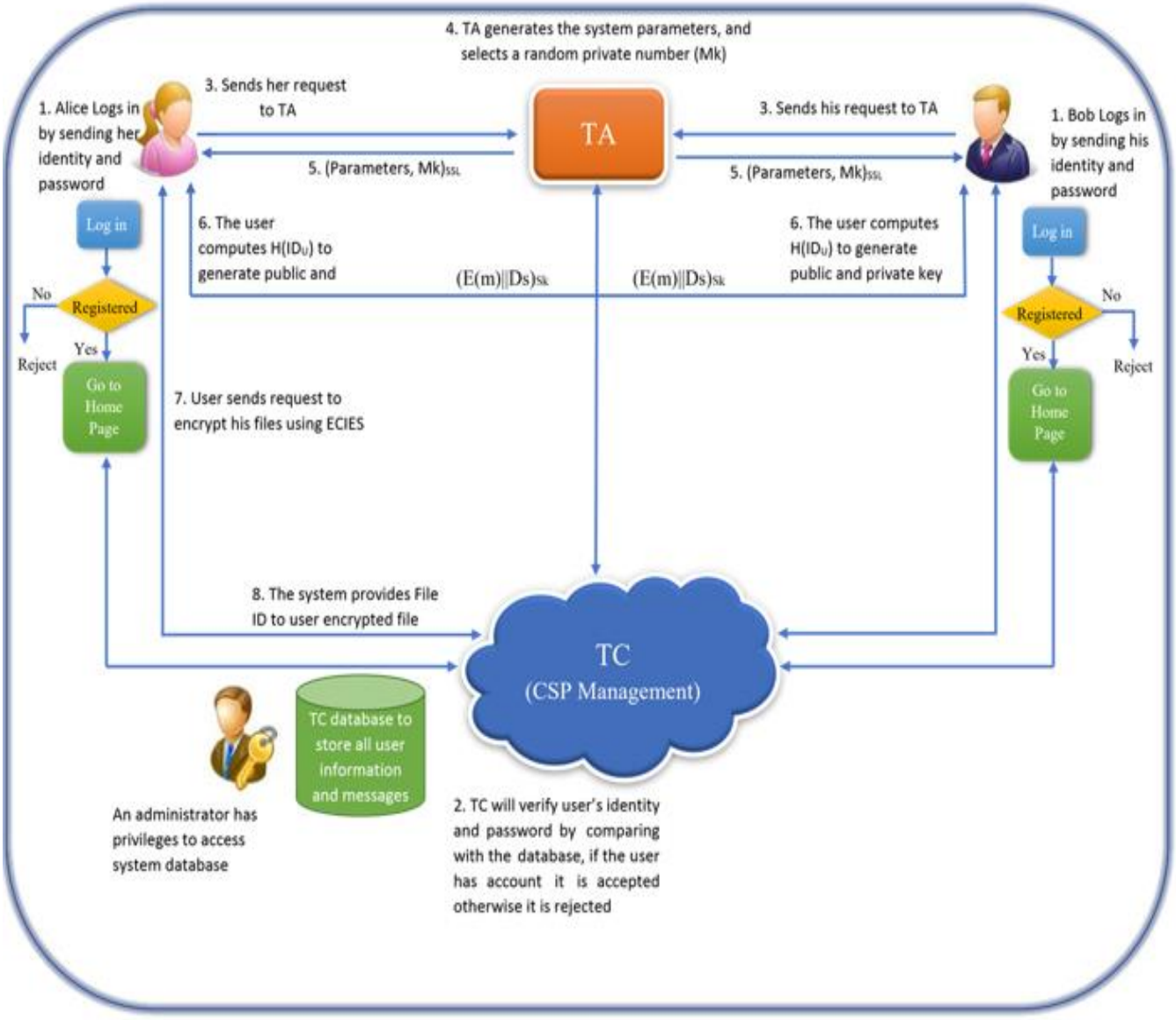


Figure 3.7 Overall system operation

Chapter 4. SYSTEM IMPLEMENTATION AND RESULTS

In this chapter, the requirements of the proposed system are introduced. This chapter also give an explanation of the interfaces of each part of the system. It also discusses the results which are obtained from the system implementation.

4.1 Implementation

The system has been implemented using a collection of web design programming languages, namely, HTML5, PhP, JavaScript, Cascading Style Sheet (CSS), and MySQL (for generating database). These platforms or languages are chosen as the most suitable for the implementation since they are considered familiar programming languages with great support of GUI. The primary implementation is done and codes are written using Java on NetBeans IDE program. Bouncy Castle, which is a Java library that complements the Java Cryptographic Extension that comes by default on NetBeans, is added as it contains the API that support implementation of ECC algorithms on NetBeans.

4.2 Results

This section presents the execution time for many cryptography operations in milliseconds. Figure (4.1) shows the execution times that are required in completing the operations of keys generation. The results show that it takes longer to generate the public key as compared to generating the private key.

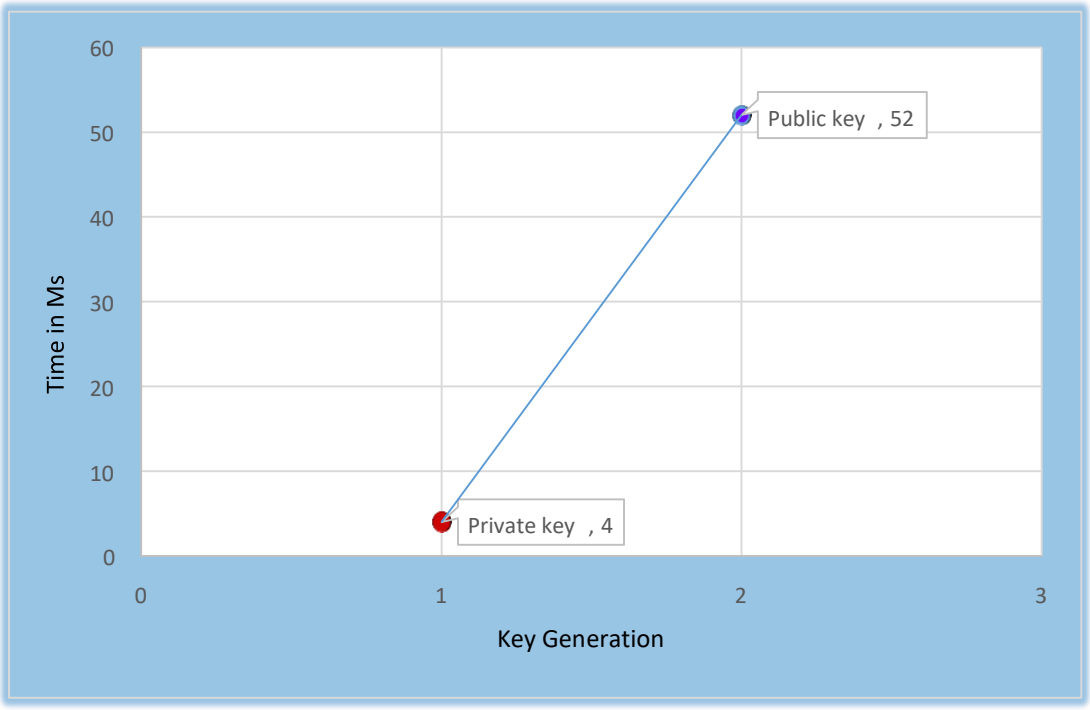
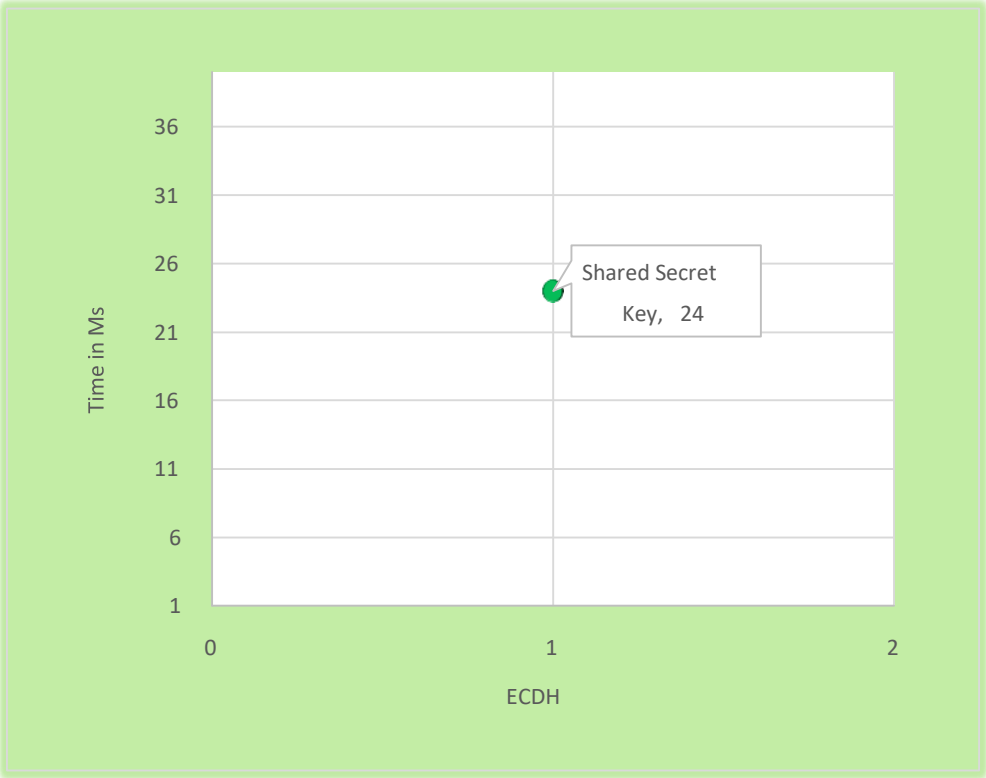


Figure 4.1 The time for Private and Public Key Generation

After each user gets his private and public keys, they will generate shared secret key by using Elliptic Curve Diffie-Hellman key exchange algorithm. Figure (4.2) shows the execution time that is required to complete the operations of shared secret key generation.



.

Figure 4.2 The time for Shared Secret Key Generation

When the user needs to encrypt a file he will use the Elliptic Curve Integrated Encryption Scheme algorithm (ECIES), and also uses it to decrypt the encrypted file and retrieve the original file. Figure (4.3) shows the execution time for encryption and decryption in ECIES algorithm, giving us the idea of how fast or the speed of encryption and decryption.

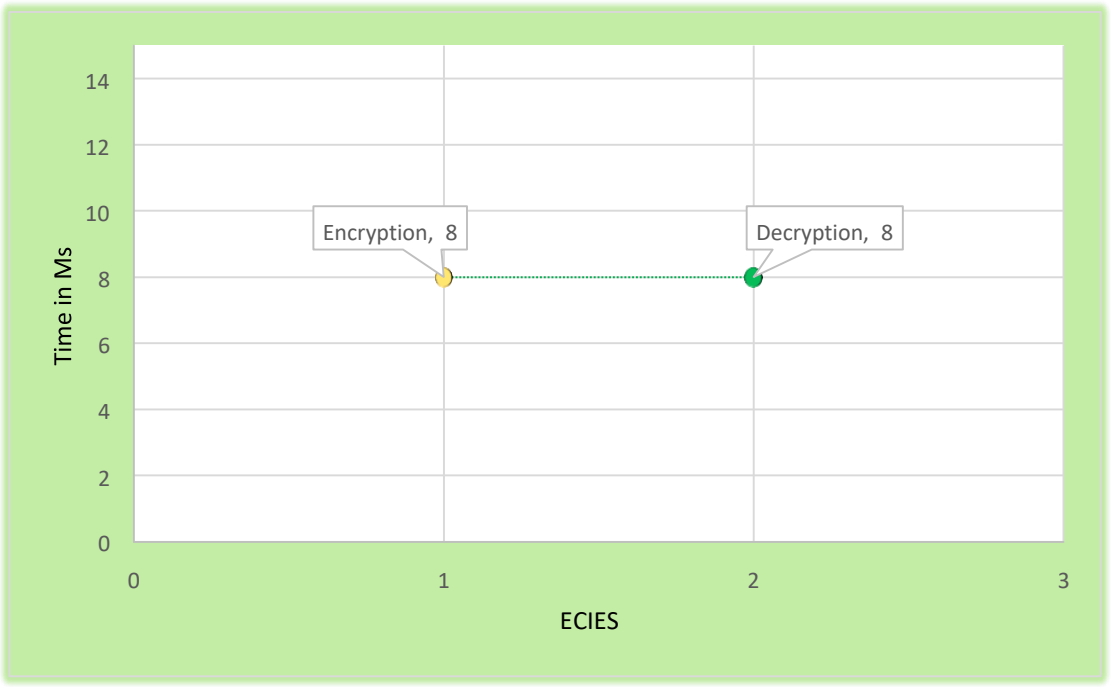


Figure 4.3 Execution time for 100KB file Encryption and Decryption

To achieve the integrity, when the receiver receives the messages he must check the integrity to ensure that the data are not altered or viewed through the transmission. This thesis uses the ECDSA to achieve the integrity. The execution time of ECDSA is illustrated in Figure (4.4).

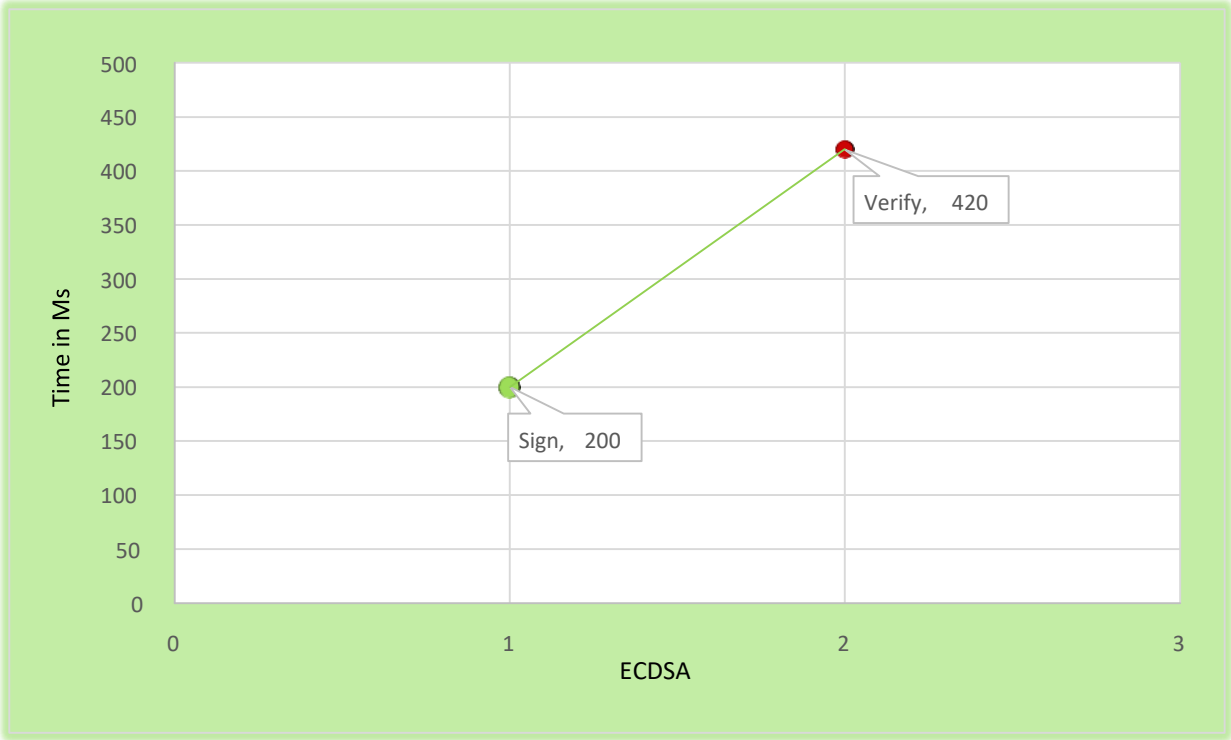


Figure 4.4 Execution Time of Signing and verifying the Message by using ECDSA

4.3 Comparison of the proposed scheme with previous works.

This section compares the proposed work with the most related previous works as presented in literature review. The following table summarizes the main distinctive points of these researchers with the work presented in this thesis.

Researchers	Comparison with Proposed System
Suli Wang et al. [30]	Used RSA algorithm with smaller sizes for file encryption and decryption, but the proposed system uses ECIES for file encryption and decryption.
Syam Kumar and Subramanian [31]	Used ECC and Sobol sequence to provide integrity and confidentiality of data. The proposed system uses ECC, ECIES to provide data confidentiality and ECDSA to provide data integrity.
Arjun Kumar et al. [32]	Used ECC to secure cloud storage, but divide the storage to two sections the first section to store user private data, and the second to store shared data and a pin number with secret key to encrypt data. The proposed system uses ECC to secure data communication and store it without dividing the storage area.
Abbas Amini [33]	Used RSA algorithm to provide data integrity and AES to achieve data confidentiality, while the proposed system uses ECC and ECIES for provide data confidentiality and ECDSA to provide data integrity.
K.Govinda and E. Sathiyamoorthy [34]	Applied the system in private cloud and used Diffe-Hellman and strong RSA for encryption, decryption, and signature, while the proposed system applies on public cloud and uses Elliptic Curve Diffe-Hellman protocol with ECC and ECIES for encryption and decryption, and ECDSA for signature.

Puneetha and Dakshayini [35]	Used ECC for data security and a hash function for digital signature. The proposed system on the other hand uses ECC and ECIES for data security and ECDSA for digital signature.
Swarnalata Bollavarapu and Bharat Gupta [36]	Used RSA, ECC and RC4 for encryption and decryption techniques, while the proposed system uses ECC and ECIES.
Nagendra Kumar, Ashok Verma and Ajay Lala [37]	Proposed a method for access control, identity, and secure data storage using RSA, while the proposed system uses ECC and ECIES.

Table 4.1 A Comparison between the Proposed Scheme and previous works.

4.4 Comparison of Time Performance Results among RSA, mediate RSA (mRSA) and ECC/ Discussion

This section compares the results of execution time of RSA algorithm and mRSA algorithm [45] and ECC in the proposed system. Performance of encryption algorithms, is evaluated considering the following parameters

1. Execution speed for private and public keys generation, encryption and decryption
2. The time consumption for signature generation and verification.

Figure (4.5) shows the test results for key generation, both private and public keys, as well as encryption and decryption. Figure (4.6) shows the test results for signing and verification.

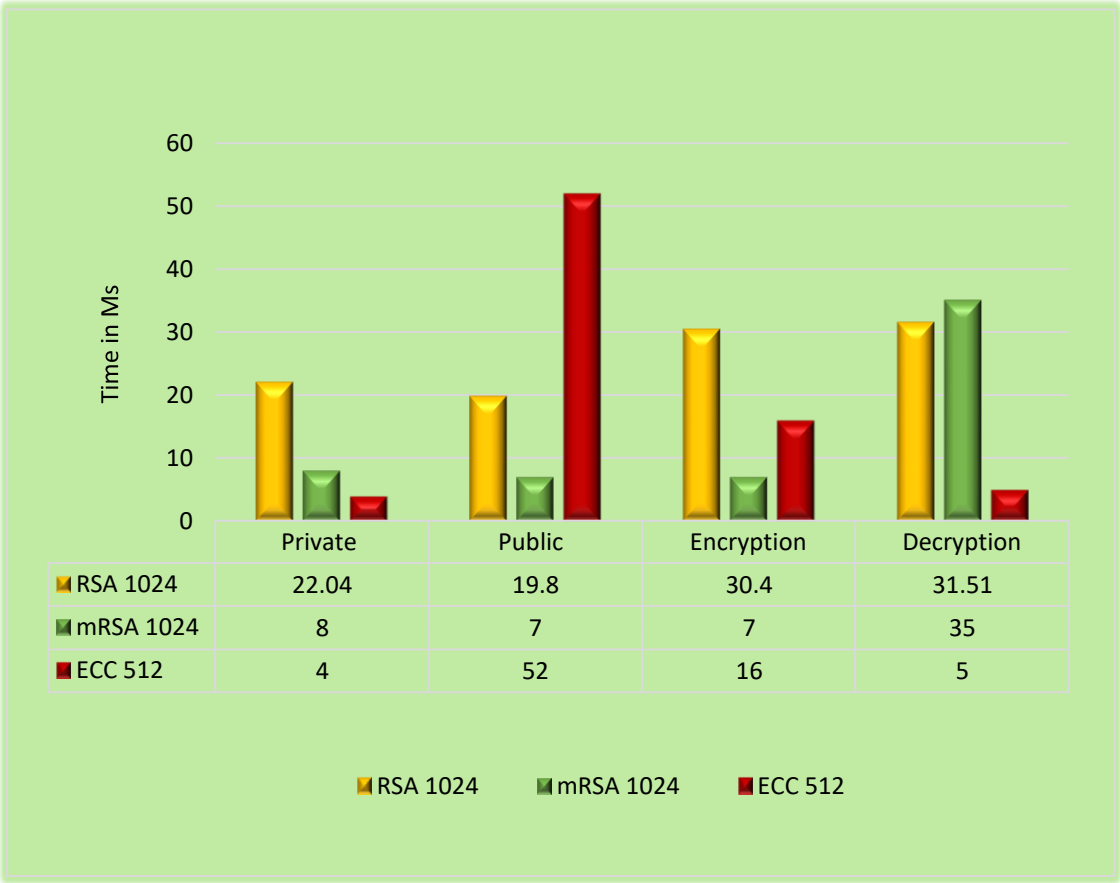


Figure 4.5 Time Performance of RSA, mRSA and ECC

Private Key generation is faster in ECC as compared to RSA and mRSA. Decryption is faster in ECC, while for encryption, it still is faster that RSA but slower than mRSA. However, this is also considering that the key size used for ECC is smaller. For ECC, the key size used is 512 Bits as compared to 1024 bits for both RSA and mRSA.

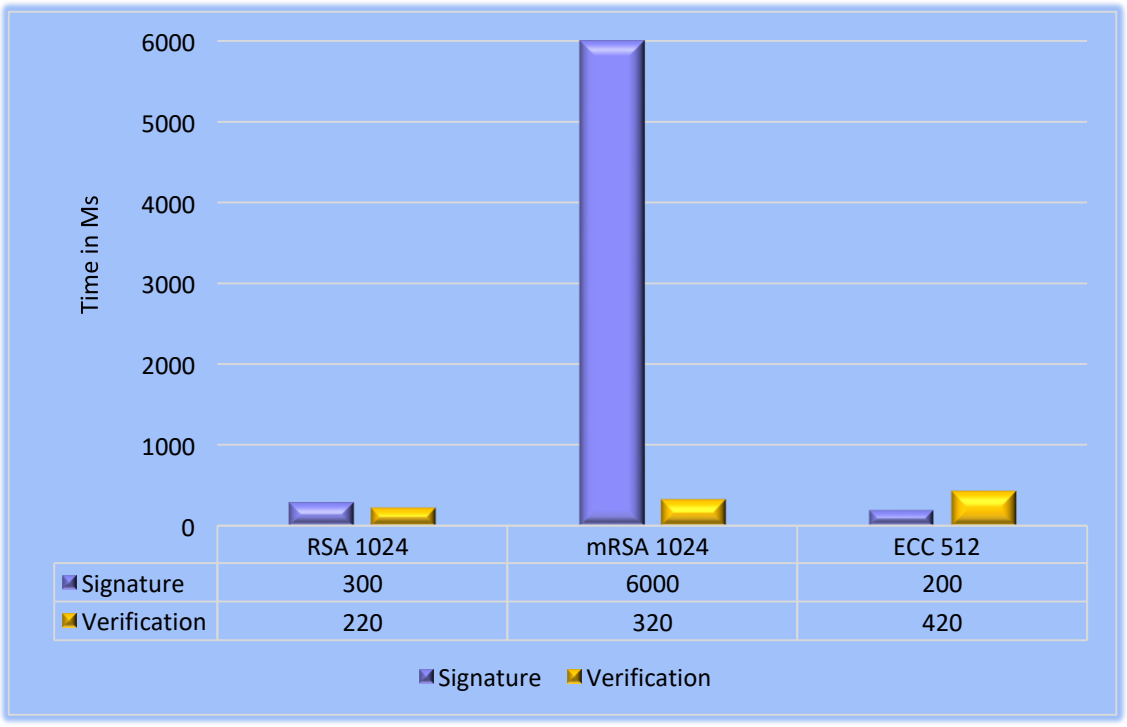


Figure 4.6 Time Performance of Signing and verification

Our proposed scheme thus comes out as an overall better technique when it comes to private key generation, encryption and decryption. It is also a winner at signature generation. It performs sub optimally when it comes to public key generation. For signature verification, the results show it performs worse as compared to RSA and mRSA. However, there is need to consider this is at half the key size of the other two. Therefore, for a larger key size, we would probably see better performance.

Chapter 5. CONCLUSIONS AND FUTURE WORK

As the final and concluding chapter of this thesis, this section presents the overall contributions of this study against the set objectives by presenting a chapter summary indicating the exact contribution. Thereafter, recommendations and suggestions for future work are discussed.

5.1 Key Contributions

In CC, it is crucial to protect personal privacy and proprietary information from unauthorized persons by keeping authorized restrictions on access and disclosure of such information. The success in achieving this goal is highly dependent on finding secure, efficient, and reliable procedures for data security and data communication. This thesis has focused the study on the problem of integrity and confidentiality of data stored in the cloud, and proposed an efficient and secure scheme using ECC. The elliptic curve discrete logarithm problem makes ECC most efficient, with smaller key size compared to the RSA algorithm. Our findings suggest that RSA key generation is significantly slower than ECC key generation. The system is based on efficient and secure combination of IBC, ECIES with TC. From the implementation of the proposed system, the following points could be concluded:

1. We can increase the security of data in the cloud by using ECIES and IBC.

2. The use of IBC eliminates the need for using digital certificates thus greater efficiency.

3. The proposed system is guarded from exposure to the problem of key escrow, because the user generates the keys unlike where the PKG is fully responsible for private key generation.

As much as RSA is the most commonly applicable cryptosystem scheme today, ECC may overtake it due to the proliferation of smaller devices and increasing security needs. Several attempts have been made at providing a secured environment for activities in the Cloud, but Elliptic Curve Cryptography (ECC) provides solutions for a secured Cloud environment with improved performance in speed, computing power and resource usage. The principal of ECC compared to

RSA is that it appears to offer equal security for a far smaller key size, thereby reducing the computation overhead. This makes it attractive for mobile applications and devices, with limited resources. The proposed method is mainly suitable for users who have less resources and limited computing capability. Having compared the proposed scheme with previously proposed schemes, we have also proved that proposed scheme is more secure and efficient.

One of the drawbacks of the system is that it is currently designed to encrypt or decrypt whole files. If a user wishes to modify a certain block of the file, then the entire file must be downloaded, decrypted, modified, then encrypted and sent back into the cloud. This definitely affects the efficiency of the system negatively.

In summary, this thesis has presented 5 chapters. Chapter 1 is the introductory chapter that gave a general overview of cloud computing, its services and a deployment models. Additionally, this chapter presented cloud security challenges with the emphasis on cryptographic protection of the data stored in the cloud. Chapter 1 also presented the research motivation, the objectives and the contributions. Chapter 2 presented a detailed review of the academic literature on cryptography in the cloud. It also presented literature on ECC, ECIES and IBC. In Chapter 3, the mathematics behind use of elliptic curves for cryptography is presented. This Chapter also presents the proposed work, and discusses the essentials parts of the proposed scheme, as well as the workings of the algorithms that make up the proposed scheme. Chapter 4 presented the implementation environment, and discusses the results obtained from the implementation of the proposed scheme.

5.2 Future Works

There are several recommendations and suggestions for the future works that can be implemented. However the one that stands out is that since Cloud Computing is expected to grow through the coming years, using a large network with only one TA may prove to be a challenge. In this case future researches might consider the use of other versions of IBC such as the hierarchical identity-based cryptography (HIBC) in place of IBC.

Appendix A

A 1.1 Bouncy Castle Provider

```
4  import cz.o2.smartbox.crypto.ecies.IESCipherGCM;
5  import cz.o2.smartbox.crypto.ecies.IESEngineGCM;
6  import org.bouncycastle.crypto.agreement.ECDHBasicAgreement;
7  import org.bouncycastle.crypto.digests.SHA256Digest;
8  import org.bouncycastle.crypto.generators.KDF2BytesGenerator;
9  import org.bouncycastle.jce.ECNamedCurveTable;
10 import org.bouncycastle.jce.interfaces.ECPrivateKey;
11 import org.bouncycastle.jce.interfaces.ECPublicKey;
   import org.bouncycastle.jce.provider.BouncyCastleProvider;
   import org.bouncycastle.jce.spec.ECNamedCurveParameterSpec;
14 import org.bouncycastle.jce.spec.ECNamedCurveSpec;
15 import org.bouncycastle.jce.spec.IESParameterSpec;
16 import org.bouncycastle.util.encoders.Base64;
17
18
19 import java.security.KeyFactory;
   import java.security.KeyPair;
   import java.security.KeyPairGenerator;
22 import java.security.SecureRandom;
23 import java.security.spec.ECPublicKeySpec;
24 import java.security.spec.PKCS8EncodedKeySpec;
```

A 1.2 Login Interface

Any user who needs to use the proposed system must sign in in this page. The user enters his identity and password, this information is encrypted by using SHA-1 then he makes sure that this account is found in MySQL database.

A 1.3 Key generation

Private Key

Code showing selection of the Elliptic curve, the generator points and the private key:

```
public function __construct(MathAdapterInterface $adapter, GeneratorPoint $generator, $secretMultiplier)
{
    $this->adapter = $adapter;
    $this->generator = $generator;
    $this->secretMultiplier = $secretMultiplier;
}
```

```
    /**
     * {@inheritDoc}
     * @see \Lib\Crypto\Key\PrivateKeyInterface::getPublicKey()
     */
    public function getPublicKey()
    {
        return new PublicKey($this->adapter, $this->generator, $this->generator->mul($this->secretMultiplier));
    }

    /**
     * {@inheritDoc}
     * @see \Lib\Crypto\Key\PrivateKeyInterface::getPoint()
     */
    public function getPoint()
    {
        return $this->generator;
    }

    /**
     * {@inheritDoc}
     * @see \Lib\Crypto\Key\PrivateKeyInterface::getCurve()
     */
    public function getCurve()
    {
        return $this->generator->getCurve();
    }
    /**
     * {@inheritDoc}
     * @see \Lib\Crypto\Key\PrivateKeyInterface::getSecret()
```

```
     */
    public function getSecret()
    {
        return $this->secretMultiplier;
    }

    /**
     * {@inheritDoc}
     * @see \Lib\Crypto\Key\PrivateKeyInterface::createExchange()
     */
    public function createExchange(MessageFactory $messageFactory, PublicKeyInterface $recipient =
null)
    {
        $exchange = new EcDH($this->adapter, $messageFactory);
        $exchange->setSenderKey($this);
        $exchange->setRecipientKey($recipient);

        return $exchange;
    }
}
```

Public Key

Code showing selection of the Elliptic curve, the generator points and the Public key:

```
public function __construct(MathAdapterInterface $adapter, GeneratorPoint $generator, PointInterface
$point)
    {
        $this->curve = $generator->getCurve();
        $this->generator = $generator;
        $this->point = $point;
```

```
        $this->adapter = $adapter;

        $n = $generator->getOrder();

        if ($n == null) {
            throw new \LogicException("Generator must have order.");
        }

        if (! $point->mul($n)->isInfinity()) {
            throw new \RuntimeException("Generator point order is bad.");
        }

        if ($adapter->cmp($point->getX(), 0) < 0 || $adapter->cmp($n, $point->getX()) <= 0
            || $adapter->cmp($point->getY(), 0) < 0 || $adapter->cmp($n, $point->getY()) <= 0
        ) {
            throw new \RuntimeException("Generator point has x and y out of range.");
        }
    }

    /**
     * {@inheritDoc}
     * @see \Lib\Crypto\Key\PublicKeyInterface::getCurve()
     */
    public function getCurve()
    {
        return $this->curve;
    }

    /**
     * {$inheritDoc}
```

```
     * @see \Lib\Crypto\Key\PublicKeyInterface::getGenerator()
     */
    public function getGenerator()
    {
        return $this->generator;
    }

    /**
     * {@inheritDoc}
     * @see \Lib\Crypto\Key\PublicKeyInterface::getPoint()
     */
    public function getPoint()
    {
        return $this->point;
    }
}
```

A.1.4 File Encryption

The user can encrypt unlimited size of data. The system generates File ID to each file, and the user must save this File ID and generate encryption key to encrypt data.

The code:

```
    /**
     * {@inheritDoc}
     * @see \Lib\Crypto\EcDH\EcDHInterface::encrypt()
     */
    public function encrypt(Message $message)
    {
        $key = hash("sha256", $this->calculateSharedKey(), true);
```

```
        $cypherText = mcrypt_encrypt(MCRYPT_RIJNDAEL_256, $key, base64_encode($message->getContent()), MCRYPT_MODE_CBC, $key);

        $message = $this->messages->ciphertext($cypherText);

        return $message;

    }
```

A 1.5 File decryption

When the user needs to decrypt the file, they use the File ID to retrieve the encryption key. They can then do the decryption and retrieve the original file.

The code:

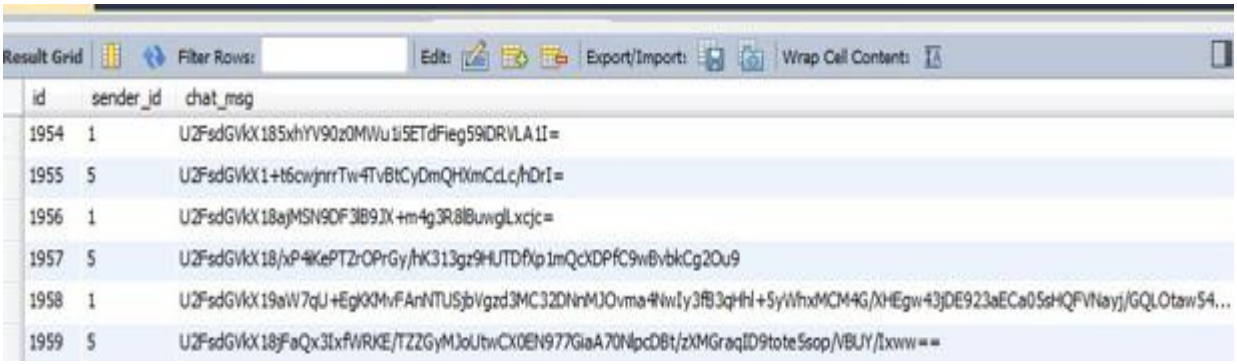
```
/**
    * {@inheritDoc}
    * @see \Lib\Crypto\EcDH\EcDHInterface::decrypt()
    */
```

```
public function decrypt(EncryptedMessage $ciphertext)
{
    $key = hash("sha256", $this->calculateSharedKey(), true);

    $clearText = base64_decode(mcrypt_decrypt(MCRYPT_RIJNDAEL_256, $key, $ciphertext->getContent(), MCRYPT_MODE_CBC, $key));
    $clearText = $this->messages->plaintext($clearText, 'sha256');
    return $clearText;
}
```

A 1.6 Database interface

The data as stored on the database, already encrypted.



References

[1] Q. Zhang, L. Cheng and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *J Internet Serv Appl,* vol. 1, pp. 7-18, 2010.

[2] P. Mell and T. Grance, "The NIST definition of Cloud Computing," *NIST special Publication,* 2011.

[3] R. Krutz and R. D. Vines, Cloud Security: A Comprehensive guide to Secure Cloud Computing, Wiley Publishing Inc, 2010.

[4] M. Iorga and S. Chokhani, Secure Cloud Computing: Cryptographic Key Management Issues and Challenges in Cloud Services, New York: Springer, 2014.

[5] G. Lewis, "Basics About Cloud Computing; Software Engineering Institute; Craig Mellon University," September 2010. [Online]. Available: https://resources.sei.cmu.edu/asset_files/WhitePaper/2010_019_001_28877.pdf. [Accessed December 2019].

[6] A. Avizienis, J.-C. Laprie, B. Randell and C. Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING,* 2004.

[7] J. Brodkin, "InfoWorld," July 2008. [Online]. Available: https://www.infoworld.com/article/2652198/gartner--seven-cloud-computing-security-risks.html.

[8] D. Mccullagh, "Dropbox confirms security glitch--no password required," 20 June 2011. [Online]. Available: https://www.cnet.com/news/dropbox-confirms-security-glitch-no-password-required/.

[9] J. Ramey and P. G. Rao, "The systematic literature review as a research genre," *IEEE International Professional Communication Conference,* pp. 1-7, 2011.

[10] Wikipedia, "Cryptography," 15 August 2019. [Online]. Available: https://en.wikipedia.org/wiki/Cryptography.

[11] M. Bellare and P. Rogaway, "Introduction," in *Introduction to Modern Cryptography*, 2005, p. 10.

[12] V. Miller, "Use of Elliptic Curves in Cryptography," in *CRYPTO*, Springer-Verlag, 1985, pp. 417-426.

[13] doubleoctopus.com, "Symmetric Key Cryptography," August 2019. [Online]. Available: https://doubleoctopus.com/security-wiki/encryption-and-cryptography/symmetric-key-cryptography/.

[14] P. Smirnoff and D. Turner, "CRYPTOMATHIC," January 2019. [Online]. Available: https://www.cryptomathic.com/news-events/blog/symmetric-key-encryption-why-where-and-how-its-used-in-banking. [Accessed July 2019].

[15] J. Andress, The Basics Of Information Security (Second edition), Elsevier Inc. All , 2014.

[16] P. Smirnoff and D. Turner , "Cryptomathic," January 2019. [Online]. Available: https://www.cryptomathic.com/news-events/blog/symmetric-key-encryption-why-where-and-how-its-used-in-banking.

[17] A. Menezes, P. Oorschot and S. Vanston, Handbook of Applied Cryptographhy, CRC Press, 2001.

[18] W. Mehuron, " Digital Signature Standard (DSS)," FIPS Publication 186-2, Gaithersburg, USA, 2000.

[19] Entrust Securing Digital Identites and Information, "Trusted Public-Key Infrastructures," Entrust, 2000.

[20] S. Choudhury, K. Bhatnagar and W. Haque, Public Key Infrastructure Implementation and Design, M&T Books, 2002.

[21] Global Institute Assurance Certifications, "Public Key Infrastructure – A Brief Overview," SANS Institute, 2005.

[22] R. K. D, V. Hu, T. P. W and S.-J. Chang, Introduction to Public Key Technology and the Federal PKI Infrastructure, National Institute of Standards and Technology(NIST), 2001.

[23] T. Edgar and D. Manz, Research Methods for Cyber Security, Elsevier Inc, 2017.

[24] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes," *Advances in Cryptology-CRYPTO,* pp. 47-53, 1985.

[25] R. Dutta, R. Barua and P. Sarkar , "Pairing-Based Cryptographic Protocols : A Survey," *International Association for Cryptologic Research(IACR),* no. 64, 2004.

[26] S. A. Abbas and A. A. Mayroosh, "Data Security for Cloud Computing based on Elliptic Curve Integrated Encryption Scheme (ECIES) and Modified Identity based Cryptography (MIBC)," *International Journal of Applied Information Systems (IJAIS) –,* vol. 10, 2016.

[27] D. Boneh and M. Franklin, "Identity-Based Encryption from the Weil Pairing," in *CRYPTO*, London, UK, 2001.

[28] J. Back, "A survey of Identity-Based Cryptography," in *Australian Unix Users Group Annual Conference*, 2004.

[29] A. Kungwani and C. Warnekar, "Techniques Practiced in Identity-based Cryptography and Applications," *International Journal of Engineering Research & Technology(IJERT),* vol. 3, no. 24, 2015.

[30] W. Suli and L. Ganlai , "File Encryption and Decryption System Based on RSA Algorithm," in *International Conference on Computational and Information Sciences*, Chengdu, China, 2011.

[31] K. Syam and R. Subramanian, "An Efficient and Secure Protocol for Ensuring Data Storage Security in Cloud Computing," *IJCSI International Journal of Computer Science Issues,* vol. 8, no. 6, 2011.

[32] K. Arjun, G. L. Byung, L. HoonJae and K. Anu, "Secure Storage and Access of Data in Cloud," in *International Conference on Information and Communication Technology Convergence (ICTC)*, Jeju Island, South Korea, 2012.

[33] A. Abbas, "Secure storage in cloud computing," Kongens Lyngby, Denmark, 2012.

[34] G. K and S. E, "Identity Anonymization and Secure Data Storage using Group Signature in Private Cloud," *Procedia Technology,* vol. 4, pp. 495-499, 2012.

[35] C. Puneetha and D. M, "Data Security in Cloud Using Elliptic Curve Cryptography," *International Journal of Innovative Research in Computer and Communication Engineering,* vol. 2, no. 5, 2014.

[36] B. Swarnalata and G. Bharat, "Data Security in Cloud Computing," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 4, no. 3, 2014.

[37] K. Nagendra , V. Ashok and L. Ajay, "Access, Identity and Secure Data Storage in Private Cloud using Digital Signature," *International Journal of Innovative Research in Computer and Communication Engineering,* vol. 2, no. 3, 2014.

[38] A. Noha MM, A. O. Fatma and F. O. Nahla, "A Hybrid Hashing Security Algorithm for Data Storage on Cloud Computing," *International Journal of Computer Science and Information Security (IJCSIS),* vol. 14, no. 4, 2016.

[39] P. T. Divya and K. S. Ajit , "A Hybrid Cryptography Algorithm forCloud Computing Security," in *International Conference on Microelectronic Devices, Circuits and Systems (ICMDCS)*, Vellore, India, 2017.

[40] R. S. Zinah, A. Zakiah, A. Nurul and R. B. Mohd, "Improved Cloud Storage Security of Using ThreeLayers Cryptography Algorithms," *International Journal of Computer Science and Information Security (IJCSIS),* vol. 6, no. 10, 2018.

[41] L. Washington, Elliptic Curves Number Theory and Cryptography, Maryland, USA: Taylor & Francis Group, 2008.

[42] M. Dubal and A. Deshmukh, "Achieving Authentication and Integrity using Elliptic Curve Cryptography Architecture," *International Journal of Computer Applications,* vol. 69, no. 24, 2013.

[43] J. Hoffstein, J. Pipher and J. Silverman, An Introduction to Mathematical Cryptography, New York: Springer, 2014.

[44] V. M. Gayoso, L. E. Hernandez and Q.-D. Araceli, "Security and Practical Considerations when Implementing the Elliptic Curve Integrated Encryption Scheme," *Cryptologia,* vol. 39, no. 3, pp. 244-269, 2015.

[45] S. Al-Janabi, S. Jassim and K. Kifayat, "Mediated IBC-Based Management System of Identity and Access in Cloud Computing," *Tikrit Journal of Engineering Sciences,* vol. 20, pp. 75-86, 2013.

[46] N. Robinson, L. Valeri, J. Cave, H. Graux and S. Creese, "The Cloud Understanding the Security,Privacy and Trust Challenges," RAND Corporation, Cambridge, 2011.

[47] Wikipedia, "Cloud Computing security," July 2019. [Online]. Available: https://en.wikipedia.org/wiki/Cloud_computing_security.

[48] Y. Z. An, Z. F. Zaaba and N. F. Samsudin, "Reviews on Security Issues and Challenges in Cloud Computing," *International Engineering Research and Innovation Symposium (IRIS),* 2016.

[49] D. Hankerson, S. Vanstone and A. Menezes, Guide to Elliptic curve cryptography, New York NY: Springer, 2004.

[50] J. Horwits and B. Lynn, "Toward Hierarchical Identity-Based Encryption," *Advances in Cryptology — Eurocrypt ,* pp. 466-481, 2002.

[51] M. Alani, Elements of Cloud Computing Security, Springer, 2016.

[52] "Clouds Computing," February 2020. [Online]. Available: https://cloudscomputing.net/cloud-architecture/.

[53] K. Harpeet, "A Novel Technique of Data Security in Cloud Computing basedon Blowfish with MD5 method," *International Journal of Advance Research, Ideas and Innovations in Technology,* vol. 3, no. 6, 2017.