APPLICATION FRAMEWORK FOR ENCRYPTING DATA FOR CLOUD TRANSMISSION USING A HOMOMORPHIC TOKEN

SHALUWAH NAZZIWA
JAN15/COMP/0560U

SUPERVISOR:
Prof. JOHN NGUBIRI

A proposal submitted to the School of Computing and Engineering in Partial fulfillment of the requirements for the award of masters of Science degree in Computing [Computer Security] of Uganda Technology and Management University [UTAMU]

DECLARATION

I declare that this dissertation has been composed solemnly by me and that it has not been submitted in whole or as part in any previous application for the award of a master's degree except where stated or otherwise by reference or acknowledgement. The work presented is entirely my own.
SIGNED

. . . .
SHALUWAH NAZZIWA
JAN15/COMP/0560U
Msc. Computing (Computer Security) Candidate

. . . .
Prof. John Ngubiri
SUPERVISOR

Contents

List of Figures

SECTION ONE

1 INTRODUCTION

1.1 Background

Data security, privacy and the process of processing data in a secure way are the most vital aspects in many if not all organizations today. Data protection involves ensuring data confidentiality, integrity and availability. Failure to protect data can lead to business losses, legal liability, and loss of company goodwill.
With many organizations, ensuring privacy and security of data and its processing has been limited by hardware costs, inefficiency in data processing, untimely delivery of data, failure in case of a disaster, uncertain of the security of data. However, with the most recent technology of cloud computing which involves hosting data and services on the internet and servers and not on the local storage elements many have managed to overcome the limitations.
Cloud computing (Cloud) is a distinct Information Technology (IT) environment designed for purposes of remotely providing scalable and measured IT resources. Cloud originated as a metaphor for the Internet which is, in essence, a network of networks providing remote access to a set of decentralized IT resources. In other words, Cloud is the delivery of on-demand computing resources ranging from applications to data centers over the internet on a pay-for-use basis.
Moving data into the cloud offers great convenience to users since they don't have to care about the complexities of direct hardware management [15]. Cloud Storage as a service where data is remotely maintained, managed, and backed up allows users to store files online, so that they can access them from any location via the Internet, it is also a user friendly type of storage where for example users can drag and drop files between the cloud storage and their local storage. It saves costs in terms of bandwidth where files can be sent via web links to recipients through email than emailing the files directly and also organizations reduce operating costs since cloud storage costs less data to store data internally and the fact that it does not require internal power to store information remotely. Above all, Cloud storage can also be used as a backup plan for organizations in cases of emergency by providing a copy of important files which is accessed through an internet connection.
Even with the above mentioned and many more benefits of cloud computing, many organizations have remained reluctant to adopt the cloud storage technology because of the challenges related to it among which is; restricted access in cases where there is no internet connections, limited bandwidth where cloud storage services have specific bandwidth allowance and cases where an organization surpasses the given allowance, the additional charges are significant, complexity of the software as it becomes difficult to manipulate files locally through multiple devices and instead you'll need to download the service on all devices[28].
The biggest challenge of cloud storage is data security and privacy [34] as indicated in Figure 1 below. Data is a vital asset for any organization and once security measures are not provided properly for data operations and transmissions then data is at high risk [27], there are many concerns with the safety, privacy and illegal access

1.jpg 1.jpg

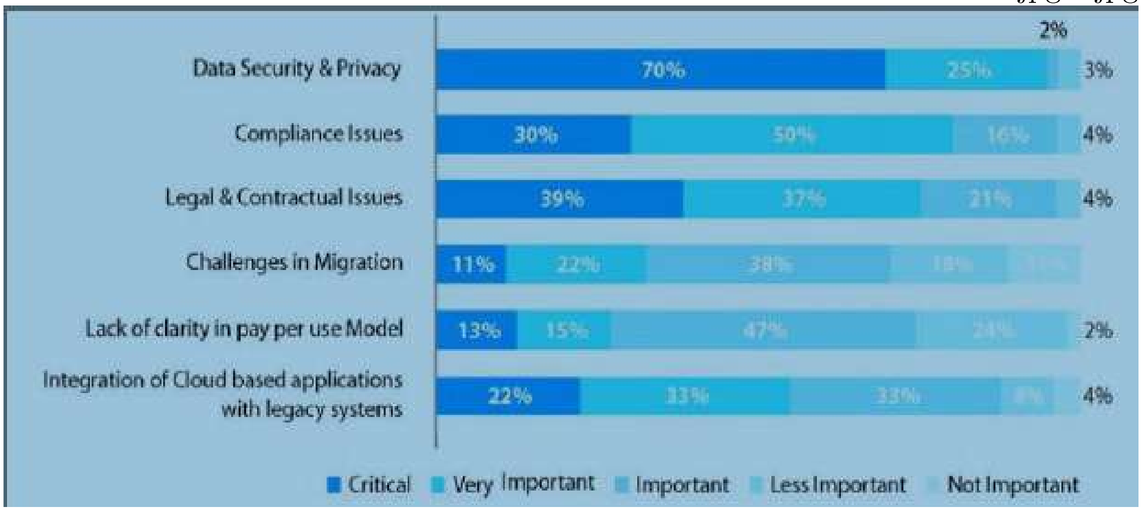


Figure 1: Data Security and Privacy - Major Inhibitor to Cloud Adoption

of important data stored remotely and this has hindered many organizations from adopting the cloud storage system[2]; however, many users / organizations are using the cloud unknowingly[27]. Among the top threats to cloud data are data loss, data breach, malicious insiders, insecure interfaces and API's, account or service hijacking, and denial of service[4].

Data losses, because of the low cost rates that the cloud offers, companies are outsourcing their entire data to cloud service providers. In this, there are high chances of losing data due to malicious attacks and sometimes due to server crashes or even the unintended deletion by the provider without having backups [37]. Among the possible solutions to data loss are using a strong API for access control, encrypting and protecting integrity of data that is in transit, analyzing data protection at run time and design time, using strong key generation, storage, destruction, and management practices, requiring the service provider to wipe the persistent media data before releasing it to the pool and specifying the back up and retention strategies.
Data breach, this is another threat to data in the cloud because cloud storage has got various users and organizations with data in the same place, any breach would expose all organizations' data. Hacking and malware are the common causes of data breaches, with 50% hacking and 49 % malware according to the "Data Breach Investigations Report" carried out in 2011[39].
Malicious insiders, these may be database administrators, employees or any other authorized to manage the data in the cloud and can decide to steal or corrupt the data for different reasons. Possible solutions for malicious insiders are conducting a comprehensive supplier assessment and making supply chain management ID stricter, defining human resources requirements, making information security and all cloud service practices more transparent and creating a process to notify when data breaches happen [2].
Insecure interfaces and Application Program Interfaces (API), API is the mode of communication used by the cloud service provider and the client and it's through it that the client is able to manage and control their data [9]. Weak and insecure

API's can lead to an illegal access of data/ resources in the cloud. Insecure interfaces and API's can be solved by analyzing the security model for interfaces of the cloud provider, making a strong access control and authentication when data is transmitted and understanding dependencies in API.
Account and service hijacking. Passwords are used to access cloud service resources, once the credentials are stolen or hijacked they can be misused and altered which is a big threat to the data since it can be stolen, altered, deleted or even sold to those who may be of much interest to it. Account or service hijacking can be solved by preventing users from sharing their credentials, using a two-factor authentication system, monitoring all activities to detect unauthorized access and understanding security policies and Service Level Agreements [4].
Cloud data could suffer from the damage on transmitting it to or from cloud data storage. Data integrity should be maintained and checked constantly in order to prove that data and computation are intact. Data integrity ensures that data is kept from unauthorized modification and any modification to the data should be detected which is not easy.

1.2 Statement of the Problem

The data stored in the cloud hosts both organizational and individual users. Data storage is the main service offered by cloud computing. Despite the fact that Cloud Service Providers (CSP) claim to offer security to the data they host, cases of security breach, information theft and data integrity violation are still experienced for different interests.
During transmission, a session can be hijacked and because data stored in the cloud sometimes needs frequent updates of insertion, deletion, modification, appending and reordering by the owner/ users, these processes like database queries, search and indexing or other application-level processing, leave the data exposed.
In public cloud environments, CSP maintain a single key for all users' data, there are high chances of forced disclosure and surveillance; Requests may come from government agencies as part of surveillance initiatives, and the CSP will not notify the data owner for permission before unlocking the data. For data at-rest (storage) a rogue admin or database administrator (DBA) can steal or leak your data.

Ensuring legal, secure and efficient access to out source data is the most important factor in cloud computing and is the basis for information management and other operations, this has not been possible for the above and many other reasons and as such a call for a framework to ensure client security of data in motion and at rest against data breaches.

1.3 Objectives of the Study

1.3.1 General objective

The general objective of this study is to develop a Framework that ensures individual persons cloud data integrity with limited or no illegal access of the data in the cloud using the homomorphic token leading to storage correctness.

1.3.2 Specific objectives

The specific objectives of the study include;

1. To analyze the security risks of data transmission and storage in the cloud.
2. To design a framework that ensures secure transmission and legal access of data stored in the cloud.
3. To test and validate the developed Framework

1.4 Significance of the Study

The main significance of the project is to provide a more effective framework ensuring security of cloud data in motion and at rest so as to reduce the vulnerabilities involved with cloud data breaches.

1.5 Scope of the Study

The scope of the framework is limited to ensuring secure data transmission to and from the cloud and some of the data stored is only accessed by the authorized persons. We will implement the framework to demonstrate homomorphism and how it can deny an illegal user access to any data that is transmitted or stored in the cloud.

SECTION TWO

2 LITERATURE REVIEW

2.1 Introduction

Cloud Computing is an Internet-based development and use of computer technology, the ever cheaper and more powerful processors, together with the software as a service (SaaS) computing architecture, are transforming data centers into pools of computing service on a huge scale. The increasing network bandwidth and reliable yet flexible network connections make it even possible that users can now subscribe high quality services from data and software that reside solely on remote data centers.
Moving data into the cloud offers great convenience to users since they don't have to care about the complexities of direct hardware management [11]. The pioneer of Cloud Computing vendors, Amazon Simple Storage Service (S3) and Amazon Elastic Compute Cloud (EC2) are both well-known examples because there unique designed that makes web-scale computing easier for developers. It has a simple web services interface that one can use to store and retrieve any amount of data, at any time, from anywhere on the web.
In cloud computing, data doesn't swirl around in a hazy cloud; it is transferred specifically to the cloud provider's data center and stored there. How it works and how this data is secure in it remains a very big question given the fact that political powers may compromise with the privacy and integrity of this data [22].

2.2 Cloud Environment Architecture

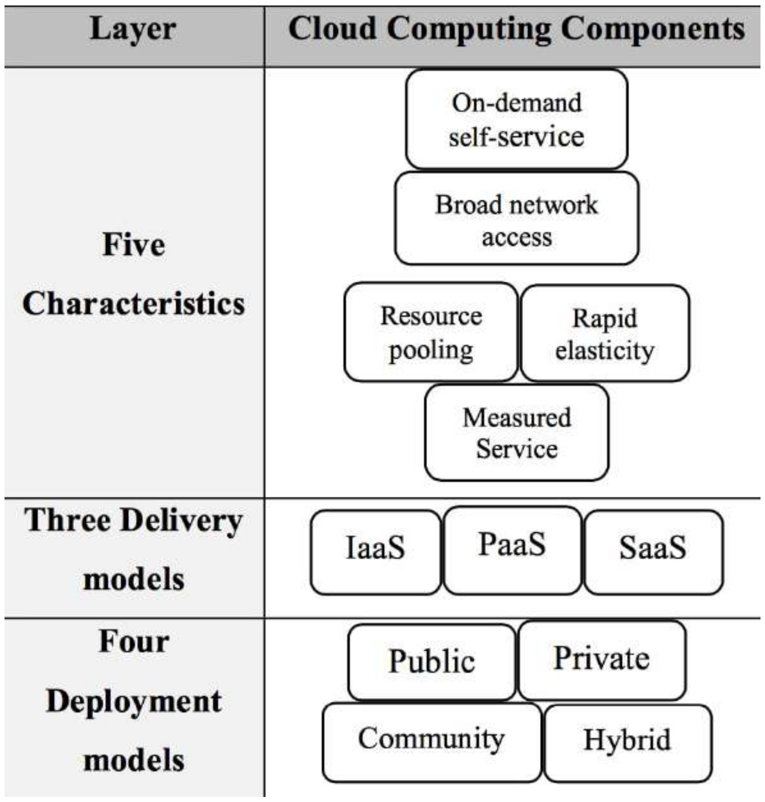
2.2.1 Cloud computing characteristics

There are basically five characteristics of cloud computing. These being on-demand self-service, where a consumer of services is provided the needed resources without human intervention and interaction with cloud provider, broad network access, which means resources can be accessed from anywhere through a standard mechanism by thin or thick client platforms such mobile phone, laptop, and desktop computer. Resource pooling, here resources are pooled in order for multi-tenants to share the resources. Rapid elasticity is where resources are dynamically increased when needed and decreased when there is no need and lastly is measured service in order to know how much is consumed. It is needed by the cloud provider in order to know how much the consumer has used to enable the billing process [37].

2.2.2 Cloud Computing Delivery or Service Models.

The models can be software as a Service (SaaS) where software and the cloud architecture are provided to the client by the cloud service provider. The client has no control over the underlying infrastructure and the physical setting of the cloud like

2.jpg 2.jpg



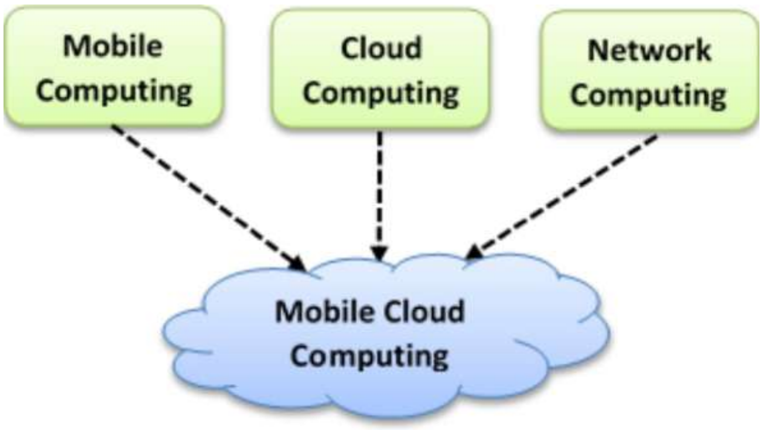
Layer	Cloud Computing Components
Five Characteristics	On-demand self-service; Broad network access; Resource pooling; Rapid elasticity; Measured Service
Three Delivery models	IaaS; PaaS; SaaS
Four Deployment models	Public; Private; Community; Hybrid

Figure 2: Cloud Environment Architecture

the network, operating system and storage. Platform as a Service (PaaS), clients can install and deploy their customized applications by using the tool offered by the cloud service provider though the physical settings remain controlled and restricted by the cloud service providers and lastly is Infrastructure as a Service (IaaS), in this model resources like processing, storage and networks are provisioned. Clients can install and use any arbitrary operating system. All these service models differ in the capabilities they offer to the consumer [18].

2.2.3 Cloud deployment models.

There are four deployment models of cloud computing as briefly discussed. Private cloud where cloud infrastructure is provided to a single organization with many consumers, the infrastructure is used exclusively by the organization. Community cloud, the cloud infrastructure is given to many organizations which forms community that shares mission, security requirements, compliance consideration or policy. The infrastructure is to be used exclusively for their uses and needs. Public cloud, this is a public provisioned model where the public uses it to satisfy their needs. It could be a government, private organization, academy owned, managed or operated cloud and lastly is the Hybrid cloud model, it comprises two or more deployment models (private, community or public) and the cloud infrastructure can be a combination of those models [39].



3.jpg 3.jpg

Figure 3: Evolution of Mobile Cloud computing.

2.2.4 Cloud data transmission

Data is transmitted in the cloud in basically three patterns. This patterns determines the performance of this data and the level under which it can easily be prone to being compromised with [2].
Enterprise to cloud is where the greatest data privacy compromise takes place. Information is transferred, typically over the open Internet, from servers in the enterprise to public cloud computing providers [18]. If one ever checked out the speed difference in downloading data from a remote website versus an intranet site, the speeds is typically lower more so when transmitting large chunks of data from the enterprise to the public cloud computing provider. Mobile Cloud computing (Mobile to cloud), this involves using the mobile as a front end and the cloud as the back end for storage and computation. Mobile cloud computing consists of mobile computing, cloud computing and network computing as indicated in figure 3 above.

The main scheme used to ensure confidentiality and integrity of mobile cloud data is the Encryption Based Scheme (EnS). In EnS, the mobile device encrypts the file and gets its hash code and the encryption key is a concatenation of the password entered by a user, file name changed to bit and file size to defend brute force attack on a cloud server since the length of the password is limited[23]. Organizations don't like to store data on mobile devices like smart phones, tablets, notebooks among others and they pretty much leave most of the data in the cloud. Only data required for an operation is typically transferred to the mobile device [17]. Lastly is the Cloud to cloud which can be either intra-cloud (such as within Amazon Web Services) or inter-cloud transfer of data (such as between AWS and Rackspace). Intra-cloud data-transfer performance is directly related to the size of the pipe within the cloud service, as well as to performance-enhancing services such as cache services that might be in use. Typically, intra-cloud data transfer is between tenants, virtual machines, or applications and data stores.
Inter-cloud data transfer is even more complex, having to deal with cloud services that may not like each other exchanging data. Moreover, the open Internet is the typical mode of transport, so the same issues arise here as with cloud-to-enterprise.

2.3 Data Encryption and Homomorphic Token

Data encryption is the translation of data into a secret code. Encryption is the most effective way to achieve data security. To read an encrypted file, one must have access to a secret key or password that enables you to decrypt it. Unencrypted data is called plain text. Encrypted data is referred to as cipher text [20].It is diagrammatically represented as in figure 4 below.

4.jpg 4.jpg

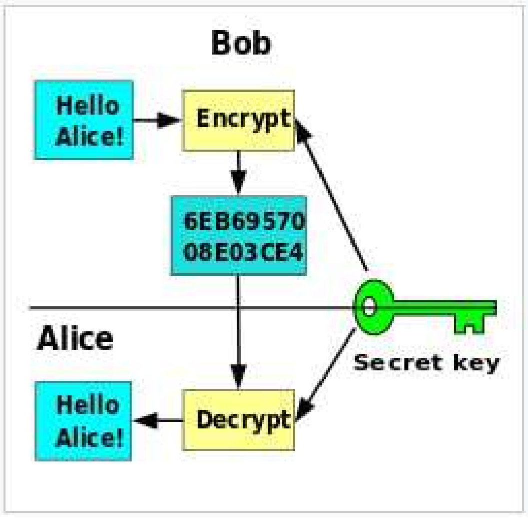
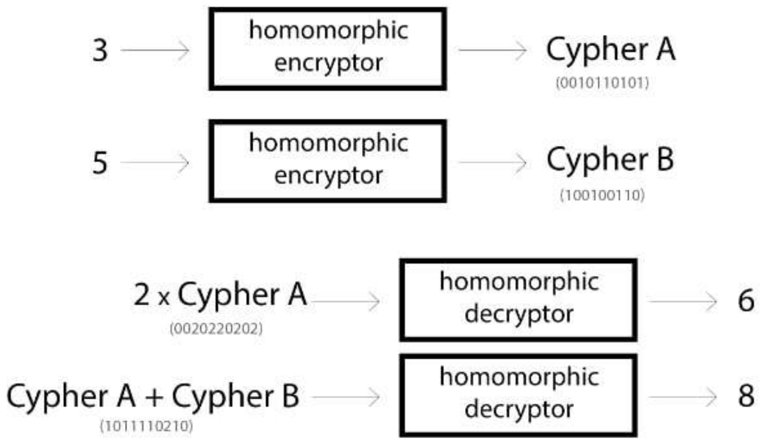


Figure 4: Encryption and Decryption of data

Homomorphic encryption (HE) on the other hand is a specific type of encryption where mathematical operations on the cipher text are equivalent to mathematical operations on the corresponding plaintext. Homomorphic encryption (HE) is desirable on account of the fact that it can simplify operations on large databases [23]. Homomorphic Encryption (HE) is a particular type of encryption that maintains certain algebraic structure between the plaintext and cipher text. One example of HE is where the product of any two cipher texts is equal to the cipher text of the sum of the two corresponding plaintexts, when all the encryptions use the same key. This is represented as: $E(m_1)xE(m_2) = E(m_1 + m_2)$ This can be demonstrated as in figure 5 on page 9 below.

HE was recognized as far back as in the 1970s, in the properties of public key cryptosystems. The potential to use these for advanced privacy protection was also recognized. This is because if the encryption is homomorphic, then fundamental factors (primitives) are enabled, such as oblivious transfer, and consequent on oblivious transfer, secure multiparty computation [8]. HE combined with threshold cryptography makes electronic elections possible [38].
To understand homomorphic encryption, it is essential to understand the algebraic meaning of the phrase. We get the term homomorphic from the algebraic term homomorphism, which refers to a mapping between two groups $(G,)$ and $(H,)$ such that $(xy) = (x)(y) forx, yGand(x), (y)H$. This notion can then be extended to rings or similar algebraic objects in the same category with multiple operations. In the context of cryptography, we consider our mapping to be $Encrypt : PC$ where $(P, +, \text{ffl})$ is the ring of plaintexts and $(C, ,)$ is the ring of cipher texts [3]. We call



5.jpg 5.jpg

Figure 5: Encryption and Decryption of data using homomorphic encryption

Encrypt a function, but this is meant in the programming sense of the word: a procedure that takes in inputs and returns a value. More precisely, Encrypt is a randomized function. It's observed that Encrypt samples randomly exist from a lattice according to some probability density function. This randomized input determines which of the many possible cipher texts a plaintext may be mapped to. So we may say that encrypt: PKC, where K is our space of randomized inputs. Intuitively, we have $Encrypt1 = Decrypt : CP$ keeping in mind that Decrypt will be many-to-one.

General cryptography Cryptography is the process of converting ordinary plain text into unintelligible text and vice-versa. It is a method of storing and transmitting data in a particular form so that only those for whom it is intended can read and process it. Cryptography not only protects data from theft or alteration, but can also be used for user authentication [21]. Earlier cryptography was effectively synonymous with encryption but nowadays cryptography is mainly based on mathematical theory and computer science practice. Modern cryptography concerns with; Confidentiality, In this case, information is aimed not be understood by anyone. Integrity, Information is not to be altered by any other third party. Non-repudiation, the Sender cannot deny his/her intentions in the transmission of the information at a later stage and Authentication, The Sender and receiver can confirm each Cryptography is used in many applications like banking transactions cards, computer passwords and e- commerce transactions with now a new end to end encryption that is in existence though with little and almost no control by the user for social media communication [35], there are different cryptographic techniques used among which are discussed below.

Symmetric-key Cryptography (Secure/Private-key encryption) Symmetric encryption is a cryptography type that uses a singular encryption key to guise an electronic message. Its data conversion uses a mathematical algorithm along with a secret key, which results in the inability to make sense out of a message. It is a two-way algorithm because the mathematical algorithm is reversed when decrypting the message along with using the same secret key [12]. The sender uses the key to encrypt plaintext and send the cipher text to the receiver. On the other side the receiver applies the same key to decrypt the message and recover the plain text.

Public-Key Cryptography (Asymmetric Encryption) Public key encryption is used to encrypt the data by using the public key. Only the one who has the private key can decrypt this data. There are many issues that make this way hard to apply in the cloud when many people need to access those files. It was the most revolutionary concept in the last 300-400 years [29]. It has disadvantages associated to it among which is key management issue and the need to get fine-grained access to file, such part of it and also this solution is not flexible and scalable because encryption and decryption is needed when a user leave the group in order to prevent him from accessing the data.

Identity-Based Encryption (IBE) The data owner encrypts the data by specifying the identity of the authorized entity to decrypt it based on the entity's identity which is matching with the one specified by the owner. In this encryption model, key exchange does not apply.

Attribute Based Encryption (ABE) Also known as Fuzzy identity-based Encryption, an identity of a user is identified by a set of attributes which generate the secret key. The data owner encrypts the data and it's only the one with attributes that overlap those specified in the cipher text that can decrypt the data [32]

Hash Functions For the hash function, no key is used in this algorithm. A fixed-length hash value is computed as per the plain text that makes it impossible for the contents of the plain text to be recovered. Hash functions are also used by many operating systems to encrypt passwords [38].

Triple DES Triple DES was designed to replace the original Data Encryption Standard (DES) algorithm, which hackers eventually learned to defeat with relative ease. At one time, Triple DES was the recommended standard and the most widely used symmetric algorithm in the industry. It uses three individual keys with 56 bits each. The total key length adds up to 168 bits, but experts would argue that 112-bits in key strength is more like it. Despite slowly being phased out, Triple DES still manages to make a dependable hardware encryption solution for financial services and other industries.

Advanced Encryption Standard (AES) AES is the algorithm trusted as the standard by the U.S. Government and numerous organizations. Although it is extremely efficient in 128-bit form, AES also uses keys of 192 and 256 bits for heavy duty encryption purposes. AES is largely considered impervious to all attacks, with the exception of brute force, which attempts to decipher messages using all possible combinations in the 128, 192, or 256-bit cipher. Still, security experts believe that AES will eventually be hailed the de facto standard for encrypting data in the private sector

2.4 Homomorphic token.

Homomorphic encryption is the conversion of data into cipher text that can be analyzed and worked with as if it were still in its original form. Homomorphic encryptions allow complex mathematical operations to be performed on encrypted data without compromising the encryption. In mathematics, homomorphic describes the transformation of one data set into another while preserving relationships between elements in both sets [8]. The term is derived from the Greek words for ”same structure.” Because the data in a homomorphic encryption scheme retains the same structure, identical mathematical operations as to whether they are performed on encrypted or decrypted data, will yield equivalent results.
The token is used in the setup phase only whereas in the time-critical online phase the cloud computes the encrypted function on encrypted data using symmetric encryption primitives only and without any interaction with other entities.

2.5 Application Framework.

Protection of mobile agents is one of the most interesting applications of homomorphic encryption. A homomorphic encryption scheme on a special non-abelian group would lead to an algebraically homomorphic cryptosystem on the finite field F_2. Since all conventional computer architectures are based on binary strings and only require multiplication and addition, such homomorphic cryptosystems would offer the possibility to encrypt a whole program so that it is still executable. Hence, it could be used to protect mobile agents against malicious hosts by encrypting them (30). The protection of mobile agents by homomorphic encryption can be used in two ways:
(i) Computing with encrypted functions.
(ii) Computing with encrypted data.
Computation with encrypted functions is a special case of protection of mobile agents. In such scenarios, a secret function is publicly evaluated in such a way that the function remains secret. Using homomorphic cryptosystems, the encrypted function can be evaluated which guarantees its privacy. Homomorphic schemes also work on encrypted data to compute publicly while maintaining the privacy of the secret data. This can be done encrypting the data in advance and then exploiting the homomorphic property to compute with encrypted data.
Multiparty computation: In multiparty computation schemes, several parties are interested in computing a common, public function on their inputs while keeping their individual inputs private. This problem belongs to the area of computing with encrypted data. Usually in multiparty computation protocols, we have a set of $n2$ players whereas in computing with encrypted data scenarios $n = 2$. Furthermore, in multi-party computation protocols, the function that should be computed is pub-

licly known, whereas in the area of computing with encrypted data it is a private input of one party.

Secret sharing scheme: In secret sharing schemes, parties share a secret so that no individual party can reconstruct the secret form the information available to it. However, if some parties cooperate with each other, they may be able to reconstruct the secret. In this scenario, the homomorphic property implies that the composition of the shares of the secret is equivalent to the shares of the composition of the secrets.

Threshold schemes: Both secret sharing schemes and the multiparty computation schemes are examples of threshold schemes. Threshold schemes can be implemented using homomorphic encryption techniques.

Zero-knowledge proofs: This is a fundamental primitive of cryptographic protocols and serves as an example of a theoretical application of homomorphic cryptosystems.

Zero-knowledge proofs are used to prove knowledge of some private information. For instance, consider the case where a user has to prove his identity to a host by logging in with her account and private password. Obviously, in such a protocol the user wants her private information (i.e., her password) to stay private and not to be leaked during the protocol operation. Zero-knowledge proofs guarantee that the protocol communicates exactly the knowledge that was intended, and no (zero) extra knowledge. Examples of zero-knowledge proofs using homomorphic property can be found in [33].

Election schemes: In election schemes, the homomorphic property provides a tool to obtain the tally given the encrypted votes without decrypting the individual votes.

Watermarking and fingerprinting schemes: Digital watermarking and fingerprinting schemes embed additional information into digital data. The homomorphic property is used to add a mark to previously encrypted data. In general, watermarks are used to identify the owner/seller of digital goods to ensure the copyright. In fingerprinting schemes, the person who buys the data should be identifiable by the merchant to ensure that data is not illegally redistributed. Further properties of such schemes can be found in [25].

Oblivious transfer: It is an interesting cryptographic primitive. Usually in a two-party 1-out of-2 oblivious transfer protocol, the first party sends a bit to the second party in such as way that the second party receives it with probability , without the first party knowing whether or not the second party received the bit. An example of such a protocol that uses the homomorphic property can be found in [8].

Commitment schemes: Commitment schemes are some fundamental cryptographic primitives. In a commitment scheme, a player makes a commitment. She is able to choose a value from some set and commit to her choice such that she can no longer change her mind. She does not have to reveal her choice although she may do so at some point later. Some commitment schemes can be efficiently implemented using homomorphic property. Lottery protocols: Usually in a cryptographic lottery, a number pointing to the winning ticket has to be jointly and randomly chosen by all participants. Using a homomorphic encryption scheme this can be realized as follows: Each player chooses a random number which she encrypts. Then using the homomorphic property the encryption of the sum of the random values can be efficiently computed. The combination of this and a threshold decryption scheme leads to the desired functionality. More details about homomorphic properties of lottery schemes can be found in [14]. Mix-nets: Mix-nets are protocols that provide anonymity for senders by collecting encrypted messages from several users. For instance, one can consider mix-nets that collect ciphertexts and output the corresponding plaintexts in a randomly permuted order. In such a scenario, privacy is achieved by requiring that the permutation that matches inputs to outputs is kept secret to anyone except the mix-net. In particular, determining a correct input/output pair, i.e., a ciphertext with corresponding plaintext, should not be more effective then guessing one at random. A desirable property to build such mix-nets is re-encryption which is achieved by using homomorphic encryption. More information about applications of homomorphic encryption in mix-nets can be found in [6].

2.6 Properties of Homomorphic Encryption Schemes

Homomorphic encryption schemes have some interesting mathematical properties. In the following, we mention some of these properties.

Re-randomizable encryption/re-encryption: Re-randomizable cryptosystems are probabilistic cryptosystems with the additional property that given the public key ke and an encryption $Ek_e(m, r)$ of a message $m \epsilon M$ under the public key k_e and a random number $r \epsilon Z$ it is possible to efficiently convert $Ek_e(m, r)$ into another encryption $Ek_e(m, r')$ that is perfectly indistinguishable from a fresh encryption of m under the public key – k_e [7]. This property is also called re-encryption. It obvious that every probabilistic homomorphic cryptosystem is re-randomizable. Without loss of generality, we assume that the cryptosystem is additively homomorphic. Given $Ek_e(m, r)$ and the public key k_e, we can compute $Ek_e(0, r'')$ for a random number r" and hence compute the following: Add$(Ek_e(m, r)$, $Ek_e(0, r'')) = Ek_e(m+0, r') = Ek_e(m, r')$ where r' is an appropriate random number. We note that this is exactly what a blinding algorithm does.

Random self-reducibility: Along with the possibility of re-encryption comes the property of random self-reducibility concerning the problem of computing the plaintext from the cipher text. A cryptosystem is called random self-reducible if any algorithm that can break a non-trivial fraction of ciphertexts can also break a random instance with significant probability [6]

Verifiable encryptions or fair encryptions: If an encryption is verifiable, it provides a mechanism to check the correctness of encrypted data without compromising on the secrecy of the data. For instance, this is useful in voting schemes to convince any observer that the encrypted name of a candidate, i.e., the encrypted vote is indeed in the list of candidates. A cryptosystem with this property that is based on homomorphic encryption can be found in [20], verifiable encryptions are also called fair encryptions.

SECTION THREE

3 METHODOLOGY

3.1 Introduction

The proposed end-to-end framework emphasizes on improving classical encryption techniques by integrating substitution cipher and transposition cipher. Both substitution and transposition techniques have used alphabet for cipher text. In the proposed algorithm, initially the plain text is converted into corresponding ASCII code value of each alphabet. In classical encryption technique, the key value ranges between 1 to 26 or key may be string (combination alphabets). But in proposed algorithm, key value range between 1 to 256. This algorithm is used in order to encrypt the data of the user in the clouds. Since the user has no control over the data after his session is logged out, the encryption key acts as the primary authentication for the user that ensures the innermost protection of the data.
This part explains how to use the a homomorphic token to ensure security of data to be transmitted and stored in the cloud, it also explains how the model is to be implemented while looking at the Software Development Life Cycle model (SDLC).

3.2 Requirements Analysis.

Aiming at coming up with the proposed prototype to implement security of cloud data, several cases will be reviewed on where secure data will find use. Cases where sensitive political information is transferred over the social media, meant for a particular person, this information will best be encrypted using the homomorphic token before it is sent. In so doing, this information will not easily be pirated. This search for this kind of related incidences will lead to the implementation of this concept.
On the secondary source of data, we will look at the documentations within the National Information Technology Authority (NITA) policy guidelines, where access to unauthorized information among others is criminal. Equally the proposed third party interface system (TPI) at National Identification Registration Authority (NIRA) is meant to give access to sensitive data to desired organizations like banks, ministries for example Education, internal affairs amongst others and yet no proper encryption policy has been defined to completely deny access to this data. This algorithm intends to help in securing any kind of data that's meant to be transmitted over the cloud.

3.3 Overview of the Approach.

The goal is to build up a repository to facilitate the data integration and sharing across cloud along with preservation of data confidentiality. For this we will be using a homomorphic token based encryption technique to provide data security on cloud data.

3.4 Design

The system will be developed using Java and ASP.NET. The system will be running on a local host. The local host will basically provide a demo application of the client. This model simplifies the client's task and put the core part of the system functions to the server to largely simplify the system development, maintenance and use. This greatly simplifies the client computer loads and reduces system maintenance, upgrade cost and effort, reducing the overall cost of ownership.

3.5 Development

The system is going to be developed to run as a local hosted system and later we intend it to be deployed on respective desktop or laptop computers so that data can be encrypted before it's transmitted over the cloud.

3.6 System and Evaluation

3.6.1 Performance Testing

Page loading time is an important part of providing a responsive user experience, and extensive web research suggests that it correlates to how long users will stay on a website and how satisfied they are with the interaction. It also directly determines the search engine ranking of this website. However, download speed is not the bottleneck. The bottleneck is the network latency, when visiting a website, only 20 % of the time it takes to display a Web page comes from downloading files. The rest of the time is spent processing HTTP requests and loading style sheets, script files and images. Thus a responsive web design will improve the speed of visiting website. Would it provide a better user experience and even recoup their economic losses?

3.6.2 The response rate

The system is intended to be on real time so that when any text that has to be encrypted is posted on the page, it will do real time conversion. This will make the system interactive for the users.

3.7 Maintenance

If the system is tested and found to be executing the intended goal, I can then deploy it for use if adopted by the Uganda community to be able to mobilize themselves without being interfered just because information leaked to the unintended persons (Police in this case).

3.8 Conclusion

Research indicates that Security and Privacy are the major issues that are needed to be countered, the greatest effort then is to develop an efficient System that can provide security and privacy at the user level and maintain the trust and intellectual property rights of the user. The method States that Encryption is one such method that can provide peace of mind to user and if the user has control over encryption and decryptions of his data, that will boost consumer confidence and attract more people to the cloud platform.

REFERENCES

1. Ahmad-Reza Sadeghi, Thomas Schneider and Marcel Winandy. (2010). Secure Outsourcing of Data and Arbitrary Computations with Lower Latency. Token-Based Cloud Computing.

2. Ali Gholami and Erwin Laure. (2015). Security and Privacy of Sensitive Data in Cloud Computing: A Survey of Recent Developments.

3. Ayantika Chatterjee and Indranilsengupta. (2016). Searching and Sorting of Fully Homomorphic Encrypted Data on Cloud.

4. Ayesha malik and Muhammad Mohsin Nazir. (2012). A Review on Security Framework for Cloud Computing Environment. Journal of Emerging Trends in Computing and Information Sciences, 390-394.

5. B.Anjani Kumar, K.Hari Prasad and C.Subash Chandra. (2013). Homomorphic Token and Distributed Erasure - Code for Cloud. International Journal for Research in Computer and Communication Technology (IJRCCT), 1069-1078.

6. C Fontaine and F Galand. (2007). A survey of Homomorphic Encryption For Non specialists. EURASIP Journal on Information Security, (pp. 1-10).

7. C Gentry and S Halevi. (2011). Implementing Gentry's Fully-Homomorphic Encryption Scheme. Annual International Conference on the Theory and Applications of Cryptographic Techniques., (pp. 129-148).

8. C Moore, M O'Neill, E O'Sullivan, Y Doroz and B Sunar. (2014). Practical Homomorphic Encryption: A Survey In Circuits and Systems (ISCAS).

9. Cong Wang, Qian Wang, Kui Ren and Wenjing Lou. (n.d.). Ensuring Data Storage Security in Cloud Computing.

10. D Naccache and J Stern. (1998). Anew Public -Key cryptosystem based on higher residues. In ACM conference on Computer and Communications Security, (pp. 59-66).

11. D.Purushothaman and Dr.S Abburu. (2012). An Approach for Data Storage Security in Cloud Computing. International Journal of Computer Science Issues (IJCSI).

12. Gamal, T. (1994). A Public-key cryptosystem and a signature scheme based on discrete logarithms. In proceedings of Crypto 84 on Advances in cryptology.

13. Gentry, C. (2009). Fully Homomorphic encryption using ideal lattices. 41st Annual ACM symposium on Theory of computing, (pp. 169-178).

14. H Delfs and H Knebl. (2007). Introduction to Cryptography: Principles and Applications.

15. J Feigenbaum and M Merritt. (1991). Discrete Mathematics and Theoretical Computer Science(DIMACS). ACM of Chapter Open Questions, Talk Abstracts and Summary of Discussions.

16. J Zhou and M Yung. (2010). Applied Cryptography and Network Security(ACNS). 8th International conference, (pp. 22-25). Beijing-China.

17. K.Saranya, I.Sibiya, M.Sasikala and D. Suvitha. (2015). Secure Storage Service Using Homomorphic Tokens and Dependable Erasure Coded Data in Cloud Computing. National Conference on Research Advances in Communication, Computation, Electrical Science and structures (NCRAccess - 2015), (pp. 10-22).

18. Kalpana Batra, Ch Sunitha and Sushil Kumar. (2013). An Effective Data Storage Security Scheme for Cloud Computing. International Journal of Innovative Research in Computer and Communication Engineering.

19. L.C Dos Santos, G.R Bilar and F.D Pereira. (2015). Implementation of the fully homomorphic encryption scheme over integers with shorter keys. 7th International Conference on New Technologies, Mobility and Security (NTMS), (pp. 1-5).

20. Liu, D. (2015). Practical Fully Homomorphic Encryption without Noise Reduction.

21. Manuel Barbosa and Pooya Farshim. (2012). Delegatable Homomorphic Encryption with Applications to Secure Outsourcing of Computation.

22. Mohit Marwaha and Rajeev Bedi. (2013). Applying Encryption Algorithm for Data Security and Privac in Cloud Computing. International Journal of Computer Science Issues.

23. O Mazonka, N G Tsoutsos and M Maniatakos. (2016). Cryptoleg: A Heterogeneous abstract machine for encrypted and unencrypted computation.

24. Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes, Advances in Cryptology Eurocrypt.

25. Pandi Priya.U and Padma Priya .R. (2015). A Novel Approach For Multi-Keyword Search In Cloud Computing Using Homomorphic Token Pre-Computation. International Journal of Innovative Science, Engineering and Technology.

26. R.L Rivest, A Shamir and L Adleman. (1978). A Method for Obtaining digital Signatures and Public-Key CryptoSystems. Communications of the ACM, (pp. 120-140).

27. R.Velumadhava Rao and K.Selvamani. (2015). Data Security Challenges and Its Solutions in Cloud Computing. International Conference on Intelligent Computing, communication and Convergence (ICCC - 2015).

28. Ramalingam Sugumar and Sharmila Banu Sheik Imam. (2015). Symmetric Encryption Algorithm to Secure Outsourced Data in Public Cloud Storage. Indian Journal of Science and Technology.

29. RL Rivest, A Shamir and L Adleman. (1978). A method of obtaining digital signatures and public-key cryptosystems. Communications of the ACM.

30. S Bogos, J Gaspoz and S Vaudenay. (2016). Cryptoanalysis of a Homomorphic Encryption Scheme.

31. S Goldwasser and S Micali. (1984). Probabilistic Encryption. Journal of Computer and System Sciences., 270-297.

32. S Halevi and V Shoup. (2014). Algorithms in Helib. International Cryptology Conference., (pp. 554-571).

33. Sen, J. (2013). Homomorphic Encryption - Theory and Application. In Theory and Practice of Cryptography and Network Security Protocols and Technologies.

34. Shahzad, F. (2014). State-of-the-art Survey on Cloud Computing Security Challenges, Approaches and Solutions. The 6th International Symposium on Application of Ad hoc and Sensor Networks (AASNET'14).

35. Smart, N. (2015). Cryptography made simpler.

36. Sneha S. Bhandarkar, Buri Chanukya, Sandhya Rani, Y.V.N. Phani Kishore. (2016). Importance of TPA and Homomorphic Token for Data Storage Security in Cloud. International Journal of Advanced Research in Computer and Communication Engineering, 343-350.

37. Sultan Aldossary, William Allen, Prince Sattam Bin. (2016). Data Security, Privacy, Availability and Integrity in Cloud Computing: Issues and Current Solutions. International Journal of Advanced Computer Science and Applications(IJACSA), 485-498.

38. Tilborg, H.C.A. Van and S Jajodia. (2014). Encyclopedia of cryptography and security. Springer Science Business Media.

39. Varsha, Amit Wadhwa and Swati Gupta. (2015). Study of Security Issues in Cloud Computing. International Journal of Computer science and Mobile Computing, (pp. 230-234).

40. W.Kuan Hon, Christopher Millard and Ian Walden. (2011). The Problem of 'Personal Data' in cloud Computing: what Information is regulated? - The Cloud of unknowing. International Data Privacy Law.