Journal of Theoretical and Applied Information Technology
15th August 2019. Vol.97. No 15

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# ENHANCING USER DATA AND VM SECURITY USING THE EFFICIENT HYBRID OF ENCRYPTING TECHNIQUES

**SAM NJUKI[1], JIANBIAO ZHANG[2], EDNA TOO[1] AND HAROLD BUKO DADYE[3]**

1. College of Computer Science and Technology, Beijing University of Technology, China
2. Professor, Beijing Key Laboratory of Trusted Computing, College of Computer Science and Technology, Beijing University of Technology, China
3. College of Computer Science and Electronic Engineering, Hunan University, China

E-mail: [1]samnjuki@emails.bjut.edu.cn, [2]zjb@bjut.edu.cn, [2]ednatoo@emails.bjut.edu.cn, [3]bukodadye@yahoo.co.uk

**ABSTRACT**

Security of data is a big concern to individuals and organizations that may be compelled to use the cloud services since some of these data are quite sensitive. The major challenges that any Cloud Computing Service Provider (CSP) must overcome are to make sure that its servers (VMs) are free from hacking by masqueraders or other attackers inside the CSPs. Therefore, there is the need for data to be encrypted and then sent to the cloud. This will ensure that users/clients sensitive data-in-transit and at rest is protected. In our new cloud framework, we implement for the user the best-fit hybrid algorithm for their data combined with an enhanced security for the VM instance. We propose the use of hybrid algorithms for VMs and user data encryption. The algorithms include RSA, AES 256-bit, ECC and SHA-256 which makes up a strong framework for speed, security, and integrity mainly for individual users. In the case of bulk data or organizations, we propose Homomorphic Encryption where the provider can operate on the data without decrypting it. We minimize the indices for fast decryption and then the decryption algorithm will require the computation of only two pairing operations, the hybrid of the standard encryption techniques and the homomorphic encryption.

**Keywords:** *Cloud Computing; Cloud Security; Homomorphic Encryption; Cryptography; Virtualization*

## 1 INTRODUCTION

Cloud computing has proved to be one of the most promising fields in information technology that has enabled storage of big data. Companies and individuals can now use high-end computing resources and infrastructure provided by cloud service providers (CSPs) such as Amazon Web Services (AWS), Google Cloud Platform, IBM Cloud and Microsoft Azure at a fee. Delivered over an internet connection, the cloud eliminates the company's data center or server[1].

The cloud system has about three components: First, we have the CSPs, who are owners who lease the computational power of the cloud and provide storage space to preserve the company or user's data. The CSPs stores and process data on behalf of its tenants. Secondly, we have client/owner/tenant who rents the space and the processing power of the cloud system. It can be an individual or an organization. Thirdly, we have a user who is registered to use the cloud for storage and computing by the tenant. In some cases, the user can also be a tenant.

Cloud computing has been possible through the concept of virtualization [2]. Computers in the cloud are configured to work simultaneously and the various applications use the collective computing power as if they are running on a physical machine. Virtualization has enabled the resource, application and computational sharing. However, with the advancement of cloud computing, other issues arise one of them is security risks. The CSPs are increasingly finding measures to address the security concern so that it can boost the customer trust of the system

In this study, the focus is on the security measures that are taken by the CSPs as well the tenant in order data security. Some of the security and privacy issues that should be addressed by the CSPs include: 1) *Integrity:* CSPs should ensure that the client's data stored in the cloud is not modified by unintended user or programs. *2)*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*Availability,* CSPs should ensure that the data and computational resources are not interfered with or made unavailable by malicious people. The user should always be able to access the service s of the cloud that the tenant/user has paid for. *3) Confidentiality,* ensures that data does not get into the hands of unauthorized persons. CSP should ensure that the data can be viewed or read by any individuals who are only authorized to access it. 4 *Access control/Authentication*.
Security and privacy issues must be addressed by both the CSPs and the owner/tenant/user. The CSPs has to build trust for its tenants and potential tenants so that they can maximize on the advantages of cloud services. By adopting the cloud model, organization/tenants/users do not have control over their data. Security and reliability being a major concern, the tenant/user/organization should play a part in ensuring that their data is secured before sending them to the cloud.

Therefore, in this work, the focus is enhancing the surety from user perspective and CSP. The CSP offer encryption of its VM instances (VM servers) using private /public key pair. The VMs are encrypted using public keys during launch time. The organization /tenant/user has to have access to the corresponding private key for them to access the cloud service. This is an alternative to the traditional way of having password-based access. The CSPs also offer encryption of files as soon as they arrive on its VM servers. However, the CSPs stores the keys for accessing the servers and files. Therefore, it is critical that user encrypts their data before sending to the cloud in order to optimize on privacy and security issues. When the user/tenant/organization encrypts their own data they store their own decryption keys. This means that even if the CSPS and any illegitimate person may see the file they cannot decrypt it since they do that have access to the key. The CSPs will only be responsible for storing the encrypted files.

Cryptography algorithms must be applied to data selectively according to the security requirement of that data which could make sense in using it. Data owners /organizations/tenants/users can select the right encryption schemes bearing in mind the required level of security and performance for the classified data.

A number of standard cryptography algorithms exist [3]. These algorithms have been applied in cloud computing for varied security solutions. From literature, many researchers have found out that combination of two or more algorithms gives better overall security[4]–[9]. In this context, cryptographic algorithms that ensure that the availability, confidentiality, and integrity are met with minimal resource consumption. They include Advanced Encryption Standard (AES) 256 bits, Elliptic Curve Cryptography (ECC), Rivest-Shamir-Adleman (RSA) and SHA 256. A hybrid of these methods is proposed so as to achieve high security.
The proposed cloud security framework is a blend of security measures. As the proposed security framework is intended to protect the stored data in the cloud environment, it encompasses three primary cryptography security measures, that is, access control, encryption, and integrity verification. These renowned security measures must be customized while using in order to provide adequate security to the data hosted in the cloud. Based on the nature of the organization and its applications, any one or combination of security measures can be concentrated exhaustively.

The objective of this paper is to ensure ensure that users/clients sensitive data-in-transit and at rest is protected. In our new cloud framework, we implement for the user the best-fit hybrid algorithm for their data combined with an enhanced security for the VM instance. We propose the use of hybrid algorithms for VMs and user data encryption. The algorithms include RSA, AES 256-bit, ECC and SHA-256 which makes up a strong framework for speed, security, and integrity mainly for individual users. In the case of bulk data or organizations, we propose Homomorphic Encryption where the provider can operate on the data without decrypting it. We minimize the indices for fast decryption and then the decryption algorithm will require the computation of only two pairing operations, the hybrid of the standard encryption techniques and the homomorphic encryption.

## 2 BACKGROUND

This section looks at some of the related work on popular and effective cryptographic algorithms that are deemed secure and possess other benefits.

### 2.1 Related Work

Research in cloud Security has attracted a lot of attention in the recent past, especially in

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

the era of big data with some of the researchers doing some tradeoffs between security and resource utilization.

### 2.1.1 Conventional Cryptographic Algorithms

From literature, most researchers focus on combining two methods[3]. The symmetric and hashing algorithms or asymmetric and hashing to provide security of data with the objective of meeting different security measures. The symmetric and hashing algorithms have been done by Bhardwaj et al. [10]. In their work, they studied asymmetric and Symmetric Algorithms Security Algorithms for Cloud Computing. Their emphasis was on symmetric algorithms such as AES, 3DES, RC6, Blowfish, and MD5. The authors analyzed the algorithms for different encryption and encoding techniques and they found AES to be a good candidate for key encryption and MD5 being faster when encoding. However, based on Cisco [https://www.cisco.com/c/en/us/about/security-center/next-generation-cryptography.html] MD5 is no longer safe as the hashing algorithm, they recommend SHA-256. A lot of focus is on using asymmetric keys and hashing techniques as evident from literature [5]–[9]. S & Subhashri [7] in their work concentrates on the integrity verification for the data. They achieved their objective using ElGamal encryption and SHA-256 hashing algorithm. On the other hand, Patel and Patel [6] proposed to develop the RSA with an application of HMAC function for hashing the key value on the cloud. The experimental results showed that RSA and digital signing algorithm is more efficient than other signature algorithm used on the cloud. Additionally, Singh et al. in their work proposed algorithm which is the hybrid approach of RSA and SHA1 help to provide new security solutions for cloud security. In their proposed work the major factor was data security and with their hybrid algorithm, they tried to meet the objective of data security [8]. Moreover, Panimalar & Subhashri [7]in their work concentrates on integrity verification methodology for outsourced data on the cloud. They combine the encrypting mechanism along with integrity verification strategy. They used an asymmetric and hashing cryptographic algorithms. These include the ElGamal encryption algorithm and SHA256 (SHA-2) hashing algorithm. It is used for ensuring data storage correctness on the untrusted server.

Other researchers have also proposed combining asymmetric and symmetric key in order to achieve maximum security. Dogra and Sharma [5]in their study proposed a combination of Elliptic Curve Diffie Hellman (ECDH), Blowfish (BF) and Particle Swarm Optimization (PSO) algorithms to try and improve the cloud security efficiency in terms of computational time and data size. ECDH generates keys required for encryption and decryption purpose. The actual encryption is then carried out using Blowfish algorithm. They recommend an experimental study to determine if their method works with less storage capacity and less computation time.

Other researchers have also combine d two asymmetric algorithms. Thiyagarajan & Ramachandrarao in their work shows that secure storage of data in the cloud is offered based on double encryption of ElGamal and Hyper-Elliptical Curve Cryptography (HECC) algorithms. ElGamal and HECC are utilized in the proposed method for encryption and decryption by giving some modification to the normal process. Modified cuckoo search (MCS) algorithm is utilized for integer selection in ElGamal cryptosystem and also for key selection in HECC it utilizes Gravitational search algorithm (GSO) [9].

Other researchers have used a combination of the three types of encryption algorithms, symmetric, asymmetry, and hashing. Abutaha & Amro [11] in their study proposed a new method for saving data in the cloud system. They use AES and RSA algorithms for securing data and connection based on different keys in encryption and decryption. They use a SHA1 algorithm to secure the hash table of data.

Some researcher has opted to use symmetric algorithms because of its resource efficiency. However, they lack in terms of providing efficient security of data. Maitri & Verma [12] In their work introduced a security mechanism using symmetric key cryptography algorithm and steganography. They proposed AES, Blowfish, RC6 and BRA algorithms for the security of the data with key sizes of 128 bit. The introduced LSB steganography technique for key information security. Additionally, Salim et al. [13] in their study show that RC6 gives better performance in terms of speed. On the other hand, asymmetric algorithms provide better security (RSA). Equally, Reddy et al. [14] in their work present algorithms to provide security in the cloud and protecting the data transmitted through

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

various secure channels by providing security using encryption. They concluded cryptographic algorithms like DES, AES, GOST 28147-89, CAST, RC6, SERPENT, and TWOFISH can be adopted for the optimization of data security in cloud computing. Bhute & Arjaria [15]work demonstrates an efficient secure user cloud framework with the help of AES and RC6 algorithms. Their results show that their approach is better in terms of using the private key, key randomization and provides the support of user to user, the user to cloud server and cloud server to the user as a comparison to the traditional approaches.

Based on the literature review a hybrid of the cryptographic techniques work better in ensuring that adequate security is provided for data and ensuring integrity, confidentiality, and authentication. However, some algorithms are slow in terms of performance and also require additional resources. RSA, for instance, has been said to provide acceptable security however it is computationally expensive to generate and store the keys. For this reason, we choose RSA for authentication of users in the cloud environment. For client's data encryption we combine three techniques ECC, AES, and SHA-256. ECC is said to provide similar security strength with fewer bits compared to RSA [16], [17]. AES[10]–[12], [14], [15] has been thought as one of the best symmetric algorithms by many researchers in terms of security and performance. The hashing algorithms is also needed to provide integrity of the keys and data. SHA-256 has been shown to be secure and better compared to MD5 [10].

### 2.1.2 Homomorphic Encryption

Most CSPs stores the data in plaintext format and the data users/owners have to ensure their data is secured by encrypting them before they are sent to the cloud for storage. However, whenever the data needs to be processed it must first be decrypted. In the era of big data this can cause performance latency due to delay in decryption. In some cases, the user has to first download the data into their own system before decryption. Even though the security of data in the cloud system is important, the computing performance of the cloud system should also be considered when choosing a security method.

Homomorphic Encryption is the encryption scheme which accepts encrypted inputs and performs blind processing to achieve data confidentiality. Homomorphic Encryption that can be applied to perform operations on encrypted data without decryption. Homomorphic Encryption is useful to transfer encrypted data in public area like cloud system as it allows operations on the cipher text, which can provide the same results after calculations as working directly on raw data[18], [19]. There are two kinds of Homomorphic encryption: Fully and partial [18]. Fully homomorphic encryption supports additional and multiplication operations on ciphertext. Partial homomorphic supports either addition or multiplication operation on the ciphertext. Homomorphic encryption is used with the purpose of securing big data and sensitive data with low latency and higher performance.

## 3 METHODOLOGY

### 3.1 Framework

Enhanced security framework for CSPs access and data in rest in the cloud environment. This security framework is a blend of security measures. The intention is to protect user/tenant/owner data stored in the cloud environment. The security measures include authentication, encryption(Confidentiality), integrity and availability. In order to provide adequate security of tenants' data in the cloud these measures have to be synchronized. Our framework intends to provide a wholesome security to the data in the cloud, environment by access control by limiting the user access i.e. only users with the passphrase and the key can access the cloud server. The encryption is then done by the user/tenant/client/owner before sending to the cloud. The user will be prompted by the CSP's open share point cloud-based platform to encrypt the data before moving it onto the cloud. The user will keep their private key and this key will be used when the user decrypts final output file/data. This ensures data confidentiality and integrity in the cloud environment as the data will only be available in encrypted form. The hashed shared secret key can then be shared with the authorized/intended recipients. The decryption can be done only with the provided hashed shared key.

User/tenant/client/owner may be an individual/organization who consume the cloud services from CSPs.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

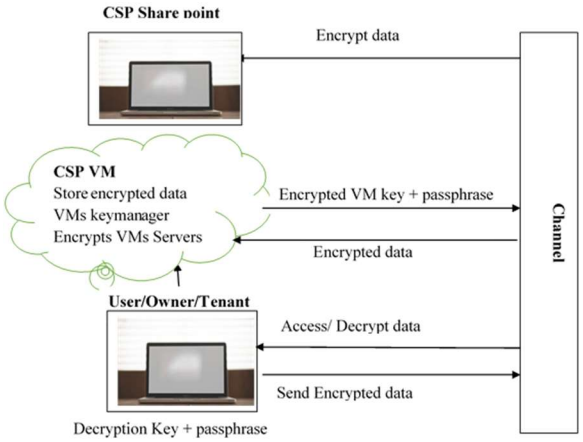


**Figure 1: Overview of the Proposed Framework**

*Figure 1* illustrates the overview of the enhanced security framework. This Framework controls the access to the cloud environment and ensures that only authorized users have access to the cloud. Additionally, it ensures that data stored in the cloud is secured. The user/tenant/owner stores only the encrypted data in the cloud.

### 3.2 Cryptographic Keys

Cryptography is based on trust, and violating that trust undermines the effectiveness of encryption technology. Users will not want to store information in either on-premises or cloud applications that they discover had been hacked into and now they include a backdoor/weakness.

#### 3.2.1 Elliptical Curve Cryptography (ECC)

Elliptical curve cryptography (ECC) is an asymmetric encryption technique based on elliptic curve theory. ECC can be used to create faster, smaller, and more efficient cryptographic keys. ECC generation of keys is through the properties of the elliptic curve equation. ECC provides strong security level with small bit keys of about 164-bit, with similar strength as RSA 1024-bit key[17]. The **reason** for choosing ECC is that it has a smaller key size, faster than RSA and its good for handheld devices and mobiles phones. use of ECC provides data confidentiality. The general equation of ECC is given as,

$$y^2 = x^3 + ax + b \qquad (1)$$

Where $A$ And $B$ Are Fixed Values From A Finite Field. A Finite Field Consists Of Integers Modulo Some Prime $P$. An Elliptic Curve Is A Set Of Points $(X,Y)$ For Which The Equation Above Is True Given Certain Chosen Numbers $A$ And $B$. An Elliptic Curve Must Have Only Points With All Coordinates Whole Numbers In The Group. If You Have A Point C On The Curve, All Multiples Of This Points Are Also On The Curve. ECC Works By Computing New Points On The Curve. The Group Operator (+) Is Used To Find The Curve Order.

We Choose A Curve With 384-Bits(Secp384r1) Which Is An Underlying Elliptic Curve (Standardized By NIST And SECG) Algorithm That Is Considered Cryptographically Secure Nowadays.

#### 3.2.2 Rivest-Shamir-Adleman (RSA)

RSA Is An Asymmetric Algorithm That Was Founded By Ron Rivest, Adi Shamir, And Len Adleman In 1978. RSA Is Used For Encrypting And Signing Data Through A Series Of Modular Multiplications. RSA Algorithms Are The Most Widely Used Public Key Algorithms, Particularly When Being Sent Over An Insecure Network Like The Internet. The RSA Key Exchange Process Is Used By Some Security Technologies To Protect Encryption Keys [6].

#### 3.2.3 Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES) is a block cipher with a block length of 128 bits. It allows three different key lengths: 128, 192, or 256 bits [20].

In this study, we propose the use of AES symmetric encryption algorithm with a key length of 256-bits. The key size needed for AES specifies the number of repetitions /rounds required to put the plaintext through the cipher and transform it into ciphertext. The encryption process consists of 14 rounds of processing for 256-bit keys. All the other rounds are identical except for the last round. AES operates on a 4 x 4 column major order matrix of bytes. AES algorithm is symmetric, meaning therefore, the same key used for encryption is also used for decryption. The choice of 256-bit key size is that longer keys are known to provide the users with stronger encryptions hence better security. However, the strength comes at the cost of performance, meaning that they will take longer to encrypt. Conversely, the 128–bits and 192-bits being shorter keys aren't as strong as the longer ones

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

(256-bits), but they provide much faster encryption times for the user.

### 3.2.4 Secure Hash Algorithm (SHA)

The objective of AES is to provide security based on its variable sizes, hashing algorithms can also be used with AES to enhance security. A more secure scheme is to store a password hash on the server. Hashing is a process where a value is calculated from text using a set of rules. Hashes are said to be better because they cannot be reversed engineered. A hash can be generated from a password, but a password cannot be generated from a hash. A hashing algorithm is said to be a mathematical function that compresses data to a fixed size. Variable fixed bit sizes exist they include 256,512, and 384.

In this study, we choose SHA with 256 bits. SHA-256 is one member of a family of cryptographic hash functions that together are known as SHA-2. The general computation for the algorithm takes a block of input data that is 512 bits and a state vector that is 256 bits in size, and it produces a modified state vector. It is a subsequent algorithm to the earlier hash algorithms MD5 and SHA-1.

According to NIST [21], the message to be hashed is first, padded with its length in such a way that the result is multiple of 512 bits long. Secondly, its parsed into 512-bit word blocks (message block) $W^1, W^2, W^3 \ldots . W^n$.

The processing of the word blocks is done one at a time. It first starts with an initial fixed hashed value $V^0$ then computing sequentially as shown in the equation:

$$\boldsymbol{V^i = V^{i-1} + F_{W^i}(V^{i-1}), \ldots\ldots\ldots\ldots V^n = V^{n-1} + F_{W^n}(V^{n-1})}$$

Where F is the function for SHA 256 compression and addition sign (+) is used for concatenation of block-phrase of mod $2^{32}$. $V^n$ is the hash of $W$.

### 3.3 Homomorphic Encryption

Fully Homomorphic Encryption (FHE) methods that were recently developed have the capability to support all kinds of functions. Many types of computations can be done on the encrypted data that is stored in the cloud without the need for any decryption. This means that operations on confidential data can now be outsourced to the cloud server keeping the secret key that can decrypt the result of the operation. {Mohammad, 2014 #10}

The distinctive technique that is used in public key cryptography is the use of asymmetric key algorithms, such that the key used to encrypt a message is not the same as the key used to decrypt it. Every user is equipped with a pair of cryptographic keys, a public key and a private key. The private key is kept secret, whereas the public key may be extensively dispersed. Usually, messages are encrypted using the recipient's public key and are only decrypted with the corresponding private key. The keys are related mathematically, but the private key cannot be feasibly derived from the public key. [16]

Key generation

1. Take two large prime numbers **p** and **q** randomly and independently of each other such that $\boldsymbol{\gcd(pq, (p-1)(q-1)) = 1}$. This method is assured if both numbers are of equal length.
2. Compute $\boldsymbol{n = pq}$ and $\boldsymbol{\lambda = lcm(p-1, q-1)}$
3. Select random integer $g$ where $\boldsymbol{g} \in \mathbb{Z}^*_{\boldsymbol{n^2}}$
4. Ensure $\boldsymbol{n}$ divides the order of $\boldsymbol{g}$ by checking the existence of the following modular multiplicative inverse: $\boldsymbol{\mu = (L(g^2 mod\ n^2))^{-1} mod\ n}$,

Where function L is defined as $\boldsymbol{L(u) = \frac{u-1}{n}}$

Note that the quotation $\frac{a}{b}$ does not denote the modular multiplication of $\boldsymbol{a}$ times the modular multiplicative inverse of $\boldsymbol{b}$ but rather the quotient of $\boldsymbol{a}$ divided by $\boldsymbol{b}$, i.e, the largest integer value $u \geq 0$ to satisfy the relation $a \geq ub$.

- The public (encryption) key is $(n, g)$
- The private (decryption) key is $(\lambda, u)$

If using $p, q$ of equivalent length, a simpler variant of the above key generation steps would be to set $g = n + 1$, $\lambda = \varphi(n)$, and $u = \varphi(\mathrm{n})^{-1} mod n$, where $\varphi(n) = (p-1)(q-1)$

**Encryption**

1. Let $m$ be a message that will be encrypted where $m \in \mathbb{Z}_{\boldsymbol{n}}$
2. Select random $r$ where $r \in \mathbb{Z}^*_{\boldsymbol{n}}$
3. Compute ciphertext as: $c = g^m . r^n mod\ n^2$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

**Decryption**

1. Let $c$ be the ciphertext to decrypt. Where $c \in \mathbb{Z}_{n^2}^*$
2. Calculate the plaintext message as: $m = L(c^{\lambda} mod\ n^2).\ \mu\ mod\ n$

Homomorphic Encryption and decryption techniques are implemented using the Paillier's algorithm using the security packages provided by Java (J2EE). These packages are also used for creation of the secrets keys.

### 3.4 Hardware And Software's

Popular virtualization technologies that have been used in our cloud computing platform are OpenStack, OpenSSL, VirtualBox, Centos 7, QEMU and J2EE.

**OpenStack-**OpenStack is an open source cloud software tool for building and management of thr cloud computing platforms for the public and the private clouds. It is mainly used for research purposes by the academician. It allows deployment of Virtual Machines (VMs) and other instances that handle different tasks for managing a cloud environment. OpenStack supports different hypervisors (QEMU, Xen, VMware or kernel-based virtual machine [KVM] for instance) and several virtualization technologies (such as bare metal or high-performance computing). OpenStack components consists of several components with a different objective. OpenStack architecture involves the following set of services: Horizon(Dashboard), Keystone(Identity), Swift (Object Storage), Neutron (Networking), Nova (compute), Glance (Images), Cinder (Block Storage) etc. access to this services are managed using roles and privileges of the users.

**QEMU-**QEMU is a generic and open source virtual machine emulator and virtualizer. QEMU operates on x86, x86-64 and PowerPC architectures and is able to emulate x86, x86-64, ARM, SPARC, PowerPC and MIPS systems. QEMU can be a bit slow therefore to increase its performance of QEMU, the Kernel Virtual Machine (KVM) component can be used.

KVM is a full virtualization solution for Linux hardware containing virtualization extensions (Intel VT or AMD-V) found on recent Linux kernels. KVM is a special operating mode of QEMU and takes advantage of hardware-assisted virtualization via the extensions Intel VT-x or AMD-V found on recent Linux kernels.

**VirtualBox-**Virtual Box is an open source virtualization software developed and maintained by Oracle. It basically provides a virtual operating system environment in another Operating system. It allows several operating systems to be installed on it as guests. It runs on major platforms and can support a number of guest OS. It has components based on QEMU but offers full virtualization rather than complete emulation. Additionally, it provides Intel's VT-x and AMD's AMD-V that support hardware-assisted virtualization.

**Centos 7-**CentOS is a free Linux based operating system. It is the most popular I Linux distributions in the hosting industry and its compatible with most Linux software. Its considered to be stable. CentOS supports the installation of OpenStack via Packstack which provides a robust and easy installation process which is easy to debug compared to others. It also supports the latest managed and stable version (Queens version).

**OpenSSL-**OpenSSL is an open source and general purpose tool that provides a wide variety of symmetric, asymmetric and hashing cryptographic keys. Its written in C language and licensed under an Apache-style license. OpenSSL has libraries for a wide range of functions such generating keys and Certificate Signing Requests, checksums, managing certificates and performing encryption/decryption. OpenSSL supports ECC key generation, encryption, and decryption.

**Ssh-Keygen**- *Ssh-keygen* is a tool for creating new authentication key pairs for SSH (Secure Shel). The authentication keys, called SSH keys, are created using the *keygen* program.The generated key pairs are used for single sign-on, hosts authenticating, and for automating logins. The SSH protocol uses public key cryptography for authenticating hosts and users.

## 4 IMPLEMENTATIONS

### 4.1 Cloud Deployment

The private cloud is deployed using OpenStack cloud (Packstack) running on Centos 7 operating system. The aim is for testing and academic research purposes. A private cloud environment is first deployed on a server and client system to interact with the deployed cloud server as depicted in *Figure 2.*

Journal of Theoretical and Applied Information Technology
15th August 2019. Vol.97. No 15

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

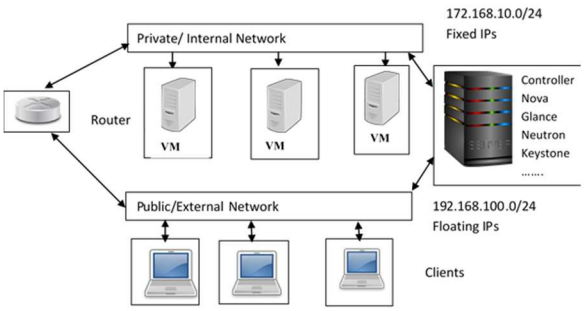


*Figure 2: OpenStack Cloud Deployment structure*

Clients (User, End users) is accessing the VM instances remotely using a floating IP address that is assigned to the VM instance. The clients can access the VM instance remotely by using their associated floating IP, *Figure 2*. They only access their part of the cloud (VM) using an IP address of that particular VM. Key management is taken over by the CSPs where a public key is stored in OpenStack (CSP) by the Key Manager while the private key on the user local workstation. Once the Key Manager (server) confirms that the two keys match, secure access is allowed.

*Figure 3* shows a login interface of the OpenStack cloud environment. The server runs all OpenStack components which include, Nova, Cinder, Horizon, swift, neutron, Keystone, and Glance. In this work, security is first achieved by encryption of VMs during launch time using RSA key pairs. The public key is stored in the cloud environment while the private keys are stored by CSPs and they are distributed to the clients. The private keys are then used by cloud clients to access the servers. The clients can then send their encrypted data to the cloud for storage. The user data are encrypted using hybrids of methods symmetric(AES), asymmetric (ECC)and hashing (SHA256) cryptographic methods.

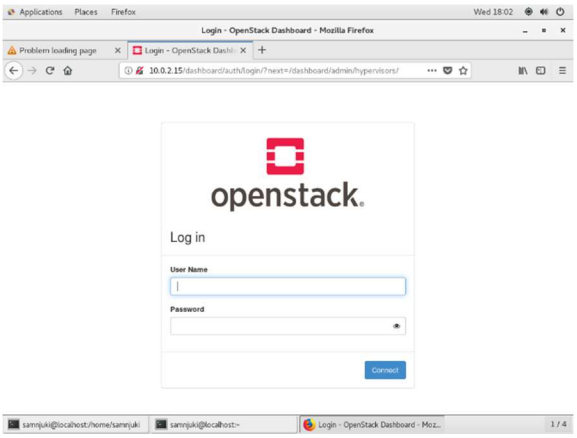


*Figure 3: OpenStack Login page*

### 4.2 VM Instance Creation and Encryption

In order to boost cloud security, CSP can set-up a security option that goes beyond password-based authentication when creating a new virtual machine (VM) instance. This can be achieved by first setting up a public/private key pairs to properly protect the VM instance at launch time. This key pairs work by keeping the public key on the server (VM instance), and the private key on your local workstation. The private key can be shared with designated clients who can then access the cloud VM server remotely. For this task, we use *ssh-keygen* to generate the public and private keys as shown in *Figure 4*. The public key is then loaded to the cloud as shown in *Figure 5* and is used to encrypt the VM instances when being launched.

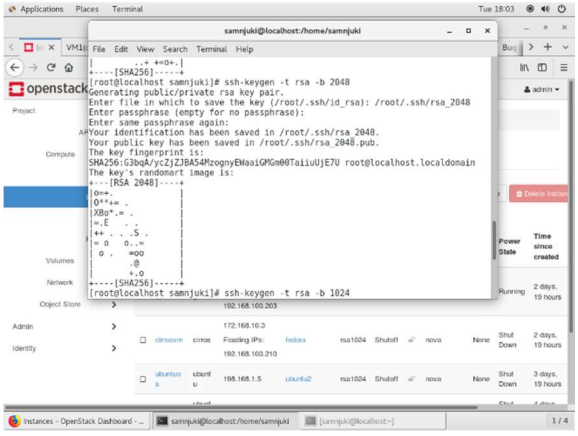


*Figure 4 : Creating Public/Private keys using SSH-KEYGEN*

Journal of Theoretical and Applied Information Technology
15th August 2019. Vol.97. No 15

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

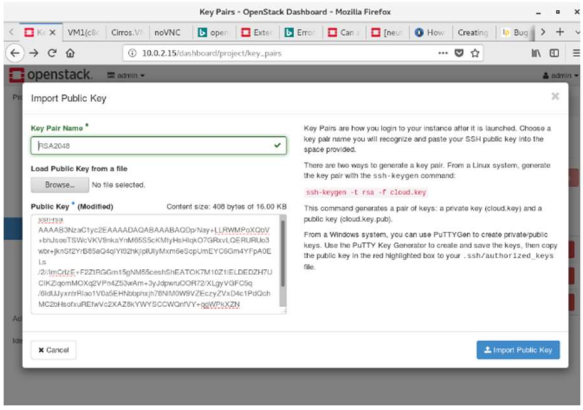


*Figure 5: Loading The Public Keys Into Openstack Cloud*

The creation of key pairs can also be achieved using OpenStack cloud environment. The OpenStack key pair process generates a unique key for each client, and securely stores this key as shown in *Figure 6*

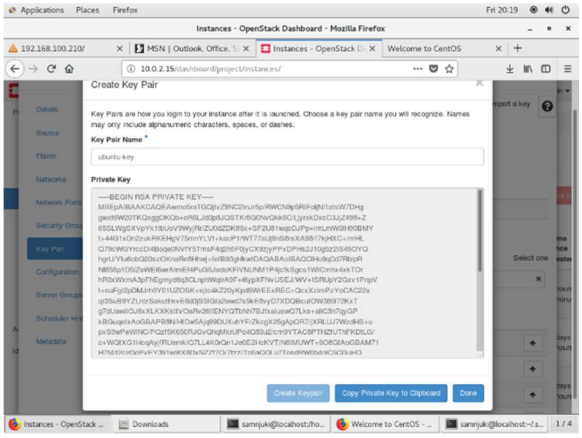


*Figure 6: Creating Key Pairs Form The Dashboard*

The keys are then stored in the key manager of the OpenStack environment. They can then be used during the launching of instances as depicted in *Figure 7.*

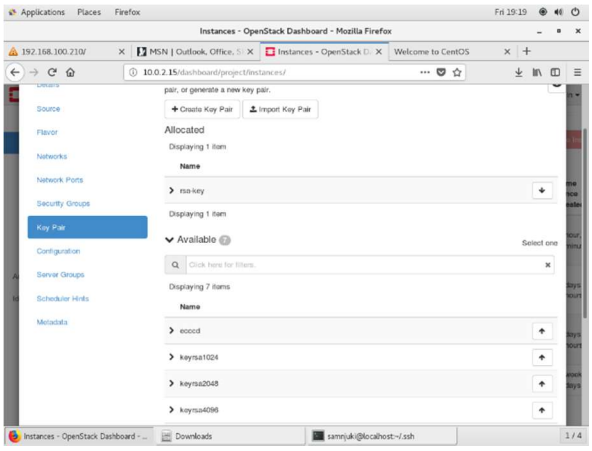


*Figure 7: List Of Public Keys On The Dashboard*

The CSP encrypts the VM instance and shares the encryption key only with the legitimate client using the Key exchange. This encryption key is passphrase protected and therefore it will be of no use if intercepted by masqueraders' attackers. This is a double protection of the VM instance. The Client will access the decryption key using the passphrase and go ahead to decrypt the VM instance ready to send their data.
To access the VM server you will then require the corresponding private key, which is used instead of the traditional password-based authentication. Once the server has verified that the two keys match, a secure connection can be made and thus, a much stronger level of security. *Figure 8* shows authentication process and access of cloud server from the client side.

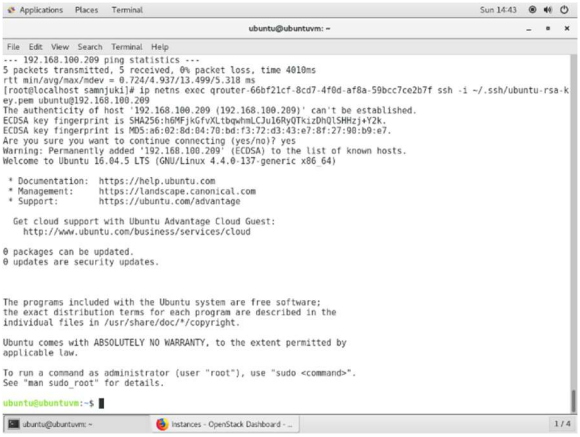


*Figure 8: Secure Access Of The VM Server (Ubuntu 16.04.5 LTS) From The Client Side Using SSH Key*

Journal of Theoretical and Applied Information Technology
15th August 2019. Vol.97. No 15

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

### 4.3 User Data Encryption and Decryption

The user will be prompted by the CSP's open share point cloud-based platform to encrypt the data before moving it onto the cloud to protect their data as per their requirement. The user will keep their private key and this key will be used when the user decrypts final output file/data.

In this paper, we have designed the best key management scheme for the data encryption that has received recent significant attention from researchers. We used a hybrid of ECC, AES-256, and SHA-256 for encryption at the local client side and for decryption on the recipient client side. This is due to their high performance, low computational cost, and small key size.

The process involves first generating temporary ECC private keys, *Figure 9 and Figure 10* shows the output of the ECC curve chosen and a sample of generated key. Secondly, generating the ECC public key from the private key. Third, using the private and the public key to derive a shared secret. Four, hash the shared secret key using SHA-256. Lastly, encrypt the file using AES-256, hashing values (SHA-256) and the derived secret key. *Figure 11* shows the output of our data in encrypted form. The user can decrypt the key using the generated shared key.

**The following procedure shows the process of securing data:**

i. Generate the ECC public key from the private key using *openssl ecparam*
ii. create public key from key generated above using *openssl ec*
iii. Use the recipient's public key to derive a shared secret using *openssl pkeyutl*
iv. Add integrity into keys/ hashing the shared keys
v. Encrypt the plaintext using *openssl enc* using the derived secret key
vi. Decrypting the encrypted file using *openssl enc –d* using the shared secret key

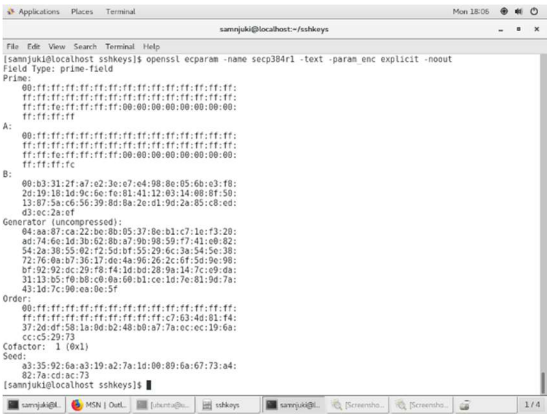


*Figure 9: Output Of The ECC*


*Figure 10: ECC Generated Key*

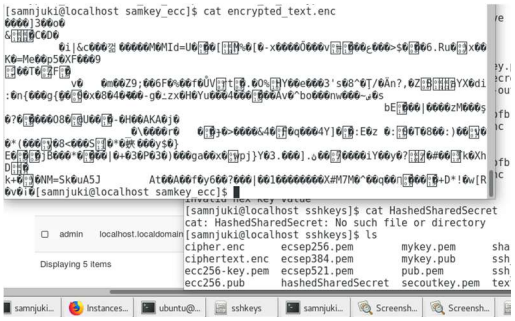


*Figure 11: Encrypted Data*

Then the data remain encrypted in the cloud and only the users accredited by the data owner can get the authority for retrieving the encrypted data. The encrypted data can be decrypted only after they are downloaded by an authorized user or the client himself. The encryption key from our experiments is accessed by using a passphrase which the client had protected their file with. This is double protection which adds to better security. The client will pass to their authorized user the passphrase to access the decryption key and the decryption key to access the data in the cloud.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

## 5 ANALYSIS

### 5.1 Performance Analysis of The Framework

We now consider the efficiency of the proposed schemes in terms of ciphertext size, private key size, and computation time for decryption and encryption. The evaluation was done on an intel core i5-4590 desktop computer with 64-bit CentOS operating system running on oracle VirtualBox with base memory of 9744Mb and dual processor with 3.3GHz.

*Table 1 Performance Analysis Of Popular Encryption Methods*

| Cryptographic method | File size (kb) | Encryption TIme | Decryption TIme | Ciphertext size( kb) |
|---|---|---|---|---|
| AES-CCM-SHA-ECC | 85 | 0.055 | 0.060 | 96 |
| AES-SHA-ECC | 85 | 0.034 | 0.036 | 85 |
| Camellia | 85 | 0.129 | 0.189 | 85 |
| RSA-2048 | 85 | 0.044 | 0.065 | 256 |
| Blowfish | 85 | 2.236 | 2.298 | 101 |

We evaluate the conventional cryptographic algorithms that are said to be secure. We measure the time it takes to encrypt and decrypt a file which is originally of size 85 kb. The performance is depicted in Table 1 and Figure 18. The blowfish method is seen to be higher as it requires a password during encryption and decryption.

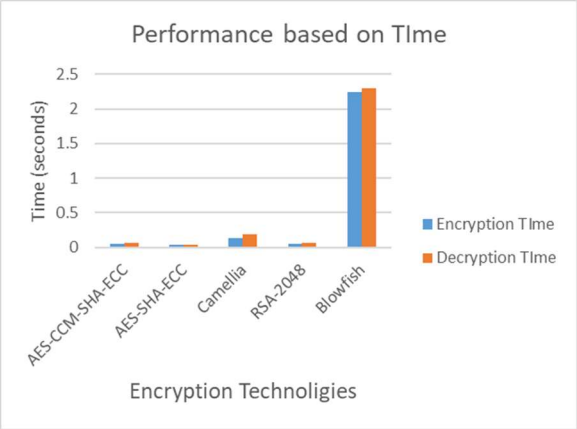


*Figure 12: Performance Of Conventional Cryptographic Techniques Based On Time*

Based on the ciphertext sizes RSA-2048 is seen to higher than the rest of the methods. Our method (AES-SHA-ECC) and camellia produces the same size of the ciphertext as the original file as shown in Figure 19. However, our method is the fastest compared to the rest of the models.

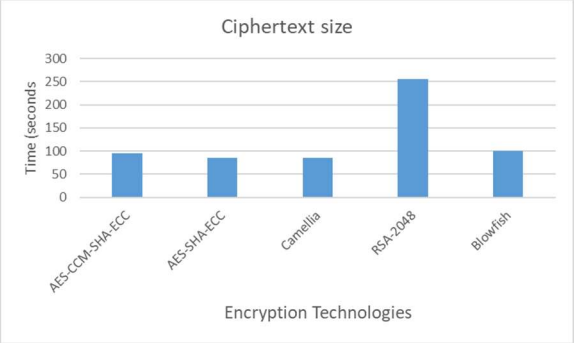


*Figure 13: The Size Of Ciphertext Based On The Conventional Cryptographic Methods*

### 5.2 Security Analysis of The Framework

In our new cloud framework, we implement for the user the best-fit hybrid algorithm for their data combined with an enhanced security for the VM instance. The CSP encrypts the VM instance using RSA-2048 bit such that one can only log into the VM using the keys that we have protected in our local machine. CSP protects the VM by encrypting it and avails the decryption key and the passphrase only to the authorized/legitimate client. The VM is encrypted using RSA with 2048 bits which is the latest acceptable algorithm that provides adequate security according to Cisco

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[https://www.cisco.com/c/en/us/about/security-center/next-generation-cryptography.html].

The cloud environment uses the RSA encryption technique for user authentication. RSA is relatively much effective in different data sizes going by the already done research. Additionally, the user encrypts their own data before being stored in the cloud. This is done using a hybrid of AES 256-bit, ECC and SHA-256 which is one of the strongest combinations of encryption protocols for speed, security, and integrity. This means that only him or an another authorized user can access them. Moreover, even if masqueraders in the intercepts the data it will be meaningless to them as it is in encrypted form. The authorized user will have to have a passphrase and decryption key for both the data and the VM. A passphrase is used to protect the encryption keys even when the keys are shared/exchanged, they are safe. This will build trust on the part of the user to the cloud environment.

Therefore, no one can read clients files unless they have the key and the passphrase to the key, which should be kept safe. Client keys are not stored in the cloud and so not even the CSPs have access to them. The system doesn't allow malicious CSP or intruders who may want to collect, sell or share your data or try to claim ownership of it, which gives you the peace of mind and trust that your data won't fall into the wrong hands.

This is a better key encryption identity to both the VM and the client data. It will act as a good root of trust for the clouds since the new model is engineered and dedicates the system to behave in an expected manner for data storage.

The framework ensures that authentication, confidentiality, availability and integrity measures are met. Authentication is achieved since the server can only be accessed by the clients who are provided with the decryption key by the CSPs. Importantly, confidentiality of sensitive data is ensured as the data is only read by the users who have the decryption key. Equally, availability is achieved *as* CSPs should ensure that the data and computational resources are not interfered with or made unavailable by malicious people. Finally, Integrity is achieved since the client's data stored in the cloud cannot be modified by unintended user or programs in the cloud environment.

This is the best well-rounded secure system for individual use and/or organizations.

## 6 CONCLUSION

In this paper, we intended to ensure ensure that users/clients sensitive data-in-transit and at rest is protected. In our new cloud framework, we implement for the user the best-fit hybrid algorithm for their data combined with an enhanced security for the VM instance. We propose the use of hybrid algorithms for VMs and user data encryption. The algorithms include RSA, AES 256-bit, ECC and SHA-256 which makes up a strong framework for speed, security, and integrity mainly for individual users. In the case of bulk data or organizations, we propose Homomorphic Encryption where the provider can operate on the data without decrypting it. We minimize the indices for fast decryption and then the decryption algorithm will require the computation of only two pairing operations, the hybrid of the standard encryption techniques and the homomorphic encryption.

In our architecture, the security of the user data does not depend on an implicit assumption of trust of the CSPs or of the service level of agreement (SLA), but instead, the user data security depends on the encryption techniques used to protect the both the VM instance and the user data.

The user gets to use the more secure hybrid encryption technique to protect their data; AES, ECC, and SHA-256. These are the latest secure cryptographic techniques from symmetric, asymmetric and hash respectively. They offer a combination of speed, security, and integrity.

The user owns the key when they encrypt their own data and also the key for their part of the VM instance. They only offer them to their legitimate/authorized user together with the passphrase for accessing it. So even the malicious CSPs or fraudsters have no access to it.

Homomorphic encryption is used in case of many users or bulk data since you don't have to download the file. We minimize the indices for fast decryption and then the decryption algorithm will require the computation of only two pairing operations, the hybrid of the standard encryption techniques and the homomorphic encryption. This reduces network latency and improves on the performance.

## ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grant 61175004 and the Specialized Research Fund for

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

the Doctoral Program of Higher Education of China under Grant 20121103110029.

## REFERENCES

[1] A. Priya and Y. K. Rana, "Design and Implementation of an Algorithm to Enhance Cloud Security," vol. 113, no. 12, pp. 41–47, 2015.

[2] S. Njuki, J. Zhang, and E. Too, "Analysis of Virtual Machine Migration Security Architectures in Cloud Computing," vol. 8, no. 10, pp. 1753–1763, 2017.

[3] S. Njuki, E. C. Too, and R. Richard, "An Evaluation on Securing Cloud Systems based on Cryptographic Key Algorithms," pp. 14–20, 2018.

[4] B. Kumar, J. Boaddh, and L. Mahawar, "A hybrid security approach based on AES and RSA for cloud data," vol. 3, no. 17, 2016.

[5] R. Dogra, "Improvement of Cloud Security Efficiency by Reducing Data Size and Computational Time Using ECDH , AES , BlowFish & PSO," vol. 8, no. 2, pp. 136–141, 2017.

[6] K. H. Patel and S. S. Patel, "Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing Alpha College of Engineering and Technology Khatraj , Kalol," vol. 4, no. 01, pp. 543–548, 2016.

[7] A. P. S and K. Subhashri, "Securing Outsourced Data On Cloud Using ElGamal Cryptosystem," pp. 53–56, 2017.

[8] S. Singh, M. T. Scholar, T. Nafis, and A. Sethi, "Cloud Computing : Security Issues & Solution," vol. 13, no. 6, pp. 1419–1429, 2017.

[9] D. Thiyagarajan and G. Ramachandrarao, "Ensuring Security for Data Storage in Cloud Computing using HECC- ElGamal Cryptosystem and GSO Optimization," *Int. J. Intell. Eng. Syst.*, vol. 10, no. 5, pp. 115–124, 2017.

[10] A. Bhardwaj, G. V. B. Subrahmanyam, V. Avasthi, and H. Sastry, "Security Algorithms for Cloud Computing," in *Procedia Computer Science*, 2016, vol. 85, pp. 535–542.

[11] M. S. Abutaha and A. A. Amro, "Using AES , RSA , SHA1 for Securing Cloud," no. April, 2014.

[12] P. V. Maitri and A. Verma, "Secure file storage in cloud computing using hybrid cryptography algorithm," *Proc. 2016 IEEE Int. Conf. Wirel. Commun. Signal Process. Networking, WiSPNET 2016*, pp. 1635–1638, 2016.

[13] P. Salim, A. Abbas, and M. Qasim, "Improving Data Storage Security in Cloud Computing Using RC6 Algorithm," vol. 19, no. 5, pp. 51–56, 2017.

[14] B. T. Reddy, K. B. Chowdappa, and S. R. Reddy, "Cloud Security using Blowfish and Key Management Encryption Algorithm," no. 6, pp. 59–62, 2015.

[15] S. Bhute and S. K. Arjaria, "An efficient AES and RC6 based cloud-user data security with attack detection mechanism," vol. 3, no. 21, 2016.

[16] N. A. Puri, A. R. Karare, and R. C. Dharmik, "Deployment of application on Cloud and enhanced data security in Cloud computing using ECC algorithm," *Proc. 2014 IEEE Int. Conf. Adv. Commun. Control Comput. Technol. ICACCCT 2014*, no. 978, pp. 1667–1671, 2015.

[17] A. O. Adetunmbi and O. S. Adewale, "Elliptic Curve Cryptography for Securing Cloud Computing Applications," vol. 66, no. 23, pp. 10–17, 2013.

[18] M. M. Potey, C. A. Dhote, and D. H. Sharma, "Homomorphic Encryption for Security of Cloud Data," *Procedia - Procedia Comput. Sci.*, vol. 79, pp. 175–181, 2016.

[19] A. Chaudhary, R. Thakur, and M. Mann, "SECURITY IN CLOUD COMPUTING BY USING HOMOMORPHIC ENCRYPTION SCHEME WITH DIFFIE-HELLMAN ALGORITHM," pp. 44–47, 2014.

[20] A. Haneena and R. Supreetha, "Protection of Password using Distributed Clustered Division and Encryption," vol. 7, no. 2, pp. 23–27, 2017.

[21] N. H. Function *et al.*, "Description of SHA-256, SHA-384 AND SHA-512," *ACM Trans. Program. Lang. Syst.*, vol. 9, no. July, p. 9, 2016.

https://www.cisco.com/c/en/us/about/security-center/next-generation-cryptography.html