International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 10 September 2023 **Revised:** 20 October 2023 **Accepted:** 30 October 2023*

Enhancing Data Security in Cloud Computing: A Comparative Analysis of Encryption Techniques

Wijdan Noaman Marzoog Al-Mukhtar
Biology Department, College of Science for Women, University of Babylon, Iraq.
Wsci.wijdan.marzoog@uobabylon.edu.iq
https://orcid.org/0009-0005-9040-4213

Abstract: This research presents a novel and efficient public key cryptosystem known as the Enhanced Schmidt Samoa (ESS) cryptosystem, proposed to safeguard the data of a single owner in cloud computing environments. Data storage is a one-time process in the cloud, while data retrieval is a frequent operation. Experimental results demonstrate that the ESS cryptosystem offers robust data confidentiality in the cloud, surpassing the security provided by traditional cryptosystems. The research also introduces a secure cloud framework designed to accommodate both individuals and organizations accessing applications and data in the cloud. While individual users may generate and share data, organizations often involve multiple users in data sharing to support their business processes. In these scenarios, multi-user data ownership and access management become critical, requiring secure sharing of cryptographic keys among the authorized users. To address these issues and ensure data confidentiality in multi-user cloud environments, the Improved Secure Cloud Data Storage Framework (ISCDSF) is introduced. This research not only enhances data security but also provides a comprehensive framework for secure data sharing in the cloud, catering to the needs of both individual users and organizations.

Keywords: Data security, Cloud computing, Encryption techniques, Comparative analysis, Key management.

1. INTRODUCTION

Cloud computing offers end users and IT organizations large-scale data storage with suitable remote access. In most personal and business contexts, there is a single owner context. You won't disclose your private information, photographs, videos, or documents with any other user. No personnel will be given access to confidential business documents, personnel information, or sensitive data. In a situation when there is only one owner, maintaining data privacy is crucial.

The main problem with cloud storage, though, is that the data cannot be administered by a single owner, or individual. Rather, the administrative controls over the data will be held by the Cloud Service Provider (CSP)[1] .

Issues with data security, such as loss of confidentiality and illegal access, are highlighted by this situation. As a result, a secure framework is required to safeguard private and corporate information. A cloud service provider will assume accountability for data security in the event that a suitable framework is put in place, replacing a single owner.

Privacy is one of the most important things. Significantly, sensitive personal data and documents belonging to a single owner must be protected when it comes to privacy. Upon obtaining sensitive data information, an intruder may encounter multiple issues. This information can only be encoded using a straight forward way[2] . The majority of the current queries, which do not employ encoded data, make this assignment perplexing.

Using a cryptographic encryption system, cloud data capacity can keep up with its confidentiality. There are two sorts of cryptography: public key cryptography and shared key cryptography. For safe data transmission and capacity, remarkable cloud service providers (CSPs) like Google, Amazon, and Microsoft use the business standard Undeniable level Encryption Standard (AES). AES, which has a 256-bit key length, is used to shield data from unwanted access Aggressors can get the accreditations and keys expected to access cloud data accepting they sort out some way to infiltrate the CSP.

Investigators' employment of public key cryptosystems is a calculated move toward ensuring the security of private information under their control[3] . Crucial to this endeavor is public key cryptography, a type of cryptography that makes use of number-theoretic ideas like discrete logarithms and factorization. safe cloud data storage and recovery, as well as safe sender-receiver communication, are two important use cases for this approach, which makes use of two separate keys: the public key and the private key.

Public key cryptography is used in the context of cloud data storage and recovery to ensure that data is encrypted and safe from prying eyes both in transit and at rest within the cloud architecture. With this configuration, data can be securely transmitted to the cloud while both the Single Owner (SO) and the Cloud Service Provider (CSP) have access to the public key. The data is encrypted, and only the SO has access to the private key necessary to decrypt it. If an adversary has access to the encrypted data or intercepts the transmission, they will still be unable to decrypt the information without the private key.

Public key cryptography is similarly effective at safeguarding the transmission of sensitive information between a sender and a receiver in the context of communication. This greatly increases the privacy and secrecy of the communication by making it difficult for eavesdroppers and other bad actors to intercept and decode the message. The public key is utilized for data encryption in the cloud, and the confidential key is utilized for data decoding by the Single Proprietor (SO). The structure is also known as a deviated cryptosystem since it makes utilization of two particular keys. The Diffie Hellman key exchange, ElGamal cryptosystem, Schmidt-Samoa cryptosystem, RSA (Rivest, Shamir Adleman), ECC (Elliptic Bend Cryptosystem), and others are a part of the praiseworthy public key cryptography systems The reports obliged by the appropriate Single Proprietor (SO) are overseen by the organized Enhanced Schmidt Samoa (ESS) public key cryptosystem in the cloud[4] . The cloud service provider handles tasks like key organization, encryption, and decoding in the continuous work. Data privacy can't be guaranteed there of psyche of a vindictive cloud service provider. That is the thing we recommend SO have the choice to execute encryption and unscrambling as well as pick the length of the key. Since the data is mixed, regardless of whether a CSP tries to hack it, they can not examine the veritable data. Confidentiality of outsourced single proprietor data can be ensured by utilizing a powerful encryption algorithm and keeping the confidential key secret.

By smoothing out the encryption and decryption times and raising the cryptanalysis complexity, this examination exertion adds to the proposition of the Enhanced Schmidt Samoa (ESS) cryptosystem for single proprietor cloud data capacity[5]. Since the general population and confidential keys are made by the estimation necessities, a beast force assault against the recommended ESS strategy would take more time to finish. The single proprietor cloud architecture involving ESS for data capacity will be more secure assuming its security strength eclipses that of SS.

1.1. Essential Characteristics of Cloud Computing

- **On-demand Self Service:** Cloud clients guess that the cloud climate will give on-request services. As per their solicitation, the service provider should concede self-service access. To let clients to pay for, demand, and use services without the help of human administrators.
- **Broad Network Access:** Cloud assets can be accessed from anyplace on the planet by means of a typical system that upholds different stages.
- **Resource Pooling:** The cloud service providers consolidated their assets and utilized different occupancy models to service various clients.
- **Rapid Elasticity:** The cloud will be versatile and expandable to satisfy the needs of buyer businesses. It is straightforward for clients to add or eliminate people, assets, programming highlights, and so on.
- **Measured Service:** The use of cloud computing assets can be monitored, made due, and detailed, giving straightforwardness to the service provider and the client of the service.

1.1.1. Classification Of Cloud Computing

The categorization of clouds based on customer service and usage includes public, private, communal, and hybrid clouds.

- **Public Cloud:** An association that claims public or outside cloud framework makes it accessible to the overall population for the compensation per-use offer of cloud services. The Windows Sky-blue services stage and Google Application Motor are two cases of public clouds.
- **Private Cloud**: A solitary association is responsible for keeping up with and controlling private or inside cloud framework. A couple of occasions of private clouds are IBM Blue Clouds, Google Application Motor, Sun Cloud, etc.
- **Community cloud**: Confidential clouds can likewise be delegated community clouds. Many ventures share cloud framework, normally with same worries. It could be shown both on and off site to an outsider or by an assortment of associations.
- **Hybrid Cloud:** There are at least two public, private, or community clouds that make up cloud framework. The objective of the half and half cloud is to give clients access to additional assets and services to address their issues.

1.1.2. Deployment Models in Cloud Computing

Services are presented by cloud providers can be assembled into three classes. Figure 1 makes sense of the cloud conveyance models.

➢ **Software as a Service (SaaS)**

The cloud client gets programming as a service. End clients get the SaaS applications by means of the web. Programming as a service (SaaS) is given through Google Applications, Salesforce.com, virtual entertainment, Gmail, and different services. SaaS benefits incorporate scalability, adaptable valuing, portability, and ease of use.

➢ **Platform as a Service (Paas)**

Stage as a Service alludes to an internet based technique for leasing network capacity, equipment, and working frameworks. The cloud client gets the stage as a service. The most notable illustration of PaaS is Microsoft Windows Sky blue and Google Application Motor. Google limits designer work and keeps up with customization and arrangement as a benefit of PaaS.

➢ **Infrastructure as a Service (IaaS)**

Building a house without any preparation involving framework as a service involves having a web association and space that can be extended or contracted, contingent upon request. Instances of IaaS clouds incorporate Amazon EC2, Go Lattice, 3tera, etc. One benefit of IaaS is that the evaluating of a cloud framework can be changed in view of utilization.

➢ **Encryption**

A communication in plaintext can be transformed into cipher text using encryption, which can then be decoded back into the original message. Both encryption and decryption involve the usage of a key and an encryption method[6]. Network security is built upon a variety of data encryption techniques. Block or stream ciphers are necessary for encryption systems to function.

The encryption algorithm and required level of security determine the length and kind of keys to be used. One key is utilized for unconventional symmetric encryption[7]. Additionally, when sending data across networks, it should be secured to prevent unauthorized users from listening in on user conversations. The security of the key becomes an issue, even if the sender and recipient can both encrypt and decrypt messages using this key.

1.1.3. Benefits

- Cost Consumption— Businesses can build their computing capacity by utilizing functional costs as opposed to capital expenditures. In addition to having a decreased entry boundary, this likewise requests less inward IT staff for framework support.
- Management — Framework upkeep is taken care of by cloud service providers, and since access is by means of APIs instead of utilization establishments on computers, support necessities are additionally limited.
- Flexibility — Businesses can rapidly grow from an unassuming sending to an enormous organization and afterward decrease it if necessary.
- Redundancy— Catastrophe recuperation and business continuity can be upheld by services that utilization a few excess sites.
- User Accessible — since frameworks are accessible inside a foundation that is accessible from all over, versatile specialists are more useful.

1.2. Encryption Techniques for Data Security in Cloud

Sensitive data is secured by the use of encryption techniques. Two distinct encryption methods are frequently employed.

➢ **Symmetric Key Encryption** uses Single key is associated with data security. Both transporter and gatherer use a comparative key to scramble and unscramble.

➢ **Asymmetric Key Encryption** uses two keys are involved. The gatherer has a secret key which is private and another key public which is disseminated to everyone.

Homomorphic alludes to the ability to transform two separate assortments of things into a similar shape or impact. Completely homomorphic encryption (FHE) intends that there are no limitations on the sort of controls that can be done[8]. The few symmetric encryption strategies — which can proficiently deal with big measures of data — are analyzed in the part that follows.

1.3. Securing Single Owner Cloud Data Using Schmidt - Samoa Cryptosystem

A public-key cryptosystem comprises of

(i) Plain text,
(ii) Cipher text,
(iii) Public key,
(iv) Private key,
(v) Encryption algorithm and
(vi) Decryption algorithm.

Public key cryptography is generally utilized for key trade, digital marks, and mystery. The Schmidt Samoa (SS) cryptosystem, one of the public key cryptosystems, depends on Katja Schmidt-created whole number factorization complexity.

Algorithm 3.1 gives the Schmidt Samoa algorithm (Katja Schmidt Samoa 2006) for guaranteeing cloud data security. It can be made sense of as follows: Two indivisible numbers, p and q, are created aimlessly. It is equivalent to the RSA and Rabin algorithms, then again, actually decryption happens all the more rapidly[9]. The process of duplicating the square of p by q yields the public key. Utilizing the LCM of (p-1) and (q-1) to get modulus, the opposite of the public key is utilized to create the confidential key. Since exponentiation requires the source to do an entire calculation, this cryptosystem causes the encryption process to become lazy. The public key is utilized to play out the encryption. The Chinese leftover portion hypothesis is applied to decode the scrambled message from the shipper utilizing the confidential key[10] .

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 10 September 2023 **Revised:** 20 October 2023 **Accepted:** 30 October 2023*

Algorithm 3.1 Schmidt Samoa Cryptosystem for single owner cloud data

```
def SS_Keygen(p, q):
    # Compute public key N
    N = p * q
    # Compute private key d
    L = lcm(p - 1, q - 1)
    d = mod_inverse(N - 1, L)
    return {"public_key": N, "private_key": d}

def SS_Encrypt(m, N):
    # Encrypt the message using public key N
    C = pow(m, N, N)
    return C

def SS_Decrypt(C, p, q, d):
    # Decrypt the message using private key d
    N = p * q
    m = pow(C, d, N)
    return m
```

1.4. Securing Single Owner Cloud Data Using Proposed Enhanced Schmidt Samoa Cryptosystem

The Schmidt Samoa cryptosystem's concerns are the primary focal point of the proposed ESS approach. General society and confidential keys created by ESS are produced utilizing four different indivisible numbers. The ESS framework is introduced utilizing four enormous indivisible numbers rather than two huge indivisible numbers, which expands the trouble level of savage power assaults and the time complexity of whole number factorization. Within the ESS cryptosystem are five modules:

i. ESS_Keygen() - key generation module is used to generate Public and Private Keys
ii. ESS_Encrypt_X() - encryption module, if public key is {N,X}
iii. ESS_Encrypt_Y() - encryption module, if public key is {N,Y}
iv. ESS_Decrypt_X() - decryption module, if public key is {N,X}
v. ESS_Decrypt_Y() - decryption module, if public key is {N,Y}

In order to facilitate communication or access to resources in the cloud, the key must be generated once and used for encryption or decryption as frequently as feasible. Figure 1.1 displays the suggested architecture for protecting data belonging to a single owner[11].

The critical matches — a public key and a confidential key — are produced by a solitary proprietor. The data should be scrambled and kept on cloud capacity by a solitary proprietor. Subsequent to getting the ciphertext from the cloud, a solitary proprietor will translate the data as indicated by the need.

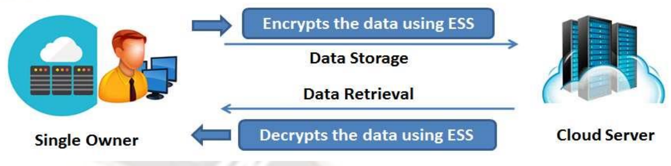

Figure 1.1 Suggested Framework for Ensuring the Security of Single Owner (SO) Data

1.4.1. ESS Key Generation Algorithm

Algorithm 3.2 presents the suggested ESS key creation (ESS_Keygen()) algorithm for single owner cloud data. It utilizes four huge prime numbers: p, q, r, and s. It has been determined that the least common multiplier is L between (p-1), (q-1), and M between (r-1), (s-1). Multiply p, q by X and r, s by Y to get the two prime numbers. The requirement that the greatest common divisors of X, L, and Y, M be 1 is required. Finding X mod L's multiplicative inverse yields the value of X'. Similar to this, Y mod M's multiplicative inverse is found in order to calculate Y'. To find Z's value, multiply X' and Y' together.

Since Z and L have exactly one common divisor, we may calculate N as the residual after dividing by L, and then find the multiplicative inverse of N modulo L, denoted by d. Therefore, N and X stand for the public keys, and d for the private one. If Z and M have the same greatest common divisor, then the process is repeated to find d, the multiplicative inverse of N modulo M. This is done in the same way as when Z and M have different greatest common divisors[12]. As a result, we set up the public keys as N and Y and keep the secret key as d. Due to the uncertainty of L and M, attempts to compromise the private key will include breaking the public key cryptosystem.

Algorithm 3.2 ESS Key Generation for single owner cloud data

Algorithm Revised ESS_Keygen for Dual-Key Cryptosystem

```
from sympy import lcm, gcd, mod_inverse, isprime, primefactors, random_prime

def ESS_Keygen(p, q, r, s):
    # Compute LCM between (p-1) and (q-1)
```

```
L = lcm(p - 1, q - 1)
# Compute LCM between (r-1) and (s-1)
M = lcm(r - 1, s - 1)

# Compute X and Y
X = p * q
Y = r * s

# Compute GCD and Inverses for X
if gcd(X, L) == 1:
    X_inverse = mod_inverse(X, L)
    Z = X_inverse
# Compute GCD and Inverses for Y
elif gcd(Y, M) == 1:
    Y_inverse = mod_inverse(Y, M)
    Z = Y_inverse

# Generate public and private keys
if gcd(Z, L) == 1:
    N = Z % L
    d = mod_inverse(N - 1, L)
    return {"public_key": N, "X": X, "private_key": d}
elif gcd(Z, M) == 1:
    N = Z % M
    d = mod_inverse(N - 1, M)
    return {"public_key": N, "Y": Y, "private_key": d}
else:
    raise ValueError("Key generation failed.")
```

1.4.2. ESS Encryption

The ESS_Keygen() strategy creates public keys, which are utilized to lay out the encryption technique for the cloud-based single-proprietor data. Figure 3.2 portrays the ESS encryption stream frame, which decides if ESS_Encrypt_X() or ESS_Encrypt_Y() ought to be utilized. At the point when the public keys are "N, X," the ESS_Encrypt_X() technique is utilized to perform Enhanced Schmidt Samoa encryption. Utilizing the given public keys N, X and the information plaintext m, the ciphertext C is produced by raising m, which is the plaintext, to the force of the modulus N, which is also a public key. On the off chance that the public keys are "N, Y," Enhanced Schmidt Samoa encryption is performed utilizing the ESS_Encrypt_Y() capability. Also, the ciphertext C is processed from the information plaintext m by raising m to the force of the public key Y over the modulus N. Both N and Y are public keys.

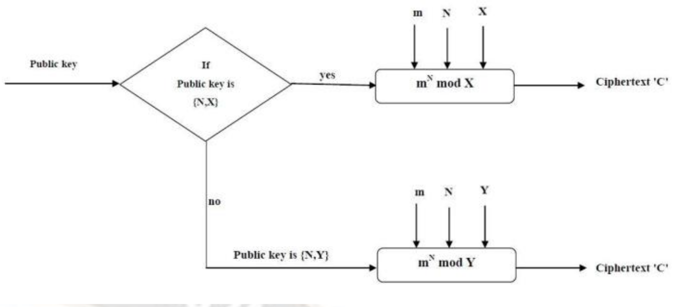

Figure 1.2 ESS encryption for single owner cloud data

The pseudo-code of ESS encryption for single owner cloud data is given in algorithm 3.3 (a), (b), as follows:

Algorithm 3.3 (a) ESS Encryption - {N,X} as Public Key

Algorithm Revised ESS_Encrypt for Dual-Key Cryptosystem

```
def ESS_Encrypt_X(m, N, X):
    # Encrypt the message using public key {N, X}
    C = pow(m, N, X)
    # Transfer the ciphertext C to the Cloud Service Provider (CSP) for storage
    return C


def ESS_Encrypt_Y(m, N, Y):
    # Encrypt the message using public key {N, Y}
    C = pow(m, N, Y)
    # Transfer the ciphertext C to the Cloud Service Provider (CSP) for storage
    return C


# Example usage
plaintext_message = 42


# Assuming keys were generated using the ESS_Keygen function
keys = ESS_Keygen(p, q, r, s)
```

```
public_key_X = {"N": keys["public_key"], "X": keys["X"]}

public_key_Y = {"N": keys["public_key"], "Y": keys["Y"]}


# Encryption using public key {N, X}

ciphertext_X = ESS_Encrypt_X(plaintext_message,
public_key_X["N"], public_key_X["X"])


# Encryption using public key {N, Y}

ciphertext_Y = ESS_Encrypt_Y(plaintext_message,
public_key_Y["N"], public_key_Y["Y"])
```

1.4.3. ESS Decryption

Based on whether ESS encryption was performed using the ESS_Encrypt_X() or ESS_Encrypt_Y() method, the appropriate ESS decryption approach is selected: ESS_Decrypt_X() or ESS_Decrypt_Y(). Figure 1.3 depicts the Enhanced Schmidt Samoa (ESS) framework's decryption procedure for cloud data belonging to a single proprietor. Methods for decrypting encrypted data are laid out in detail within this framework, with special emphasis on the conditional decision-making process that determines which decryption method is used in response to a given encryption method employed inside an ESS system.

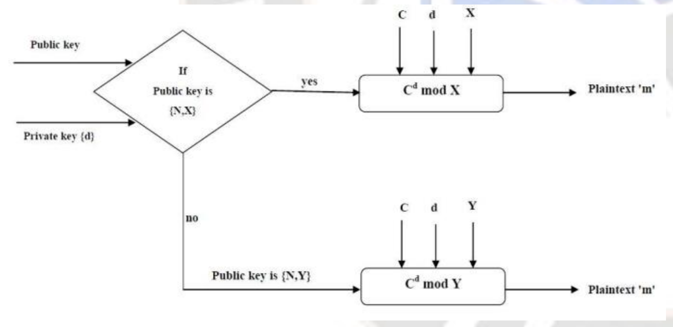

Figure 1.3 Decryption Process for Single Owner Cloud Data Using the ESS Cryptosystem

Certainly, the algorithm details the steps required to decrypt ESS for cloud data that belongs to a single owner. Depending on whether the public key N, X> or N, Y> was used for ESS encryption, two distinct decryption capabilities, ESS_Decrypt_X() and ESS_Decrypt_Y(), are described for the ESS decryption protocol. The ciphertext 'C', the public key 'N, X', and the confidential key 'd' are the contributions to the ESS_Decrypt_X() capability, which unscrambles the message utilizing the condition: ciphertext 'C' raised to the force of private key 'd' modulo public key 'X'. This process creates the plaintext'm'. The ESS_Decrypt_Y() capability utilizes a similar condition where the ciphertext 'C' raised to the force of the confidential key 'd' modulo the public key 'Y' to work out the plaintext'm', and it acknowledges similar contributions of the ciphertext 'C,' the public key 'N, Y,' and the confidential key 'd'. Enhanced security for cloud data management is provided by the algorithm, which displays the ESS decryption process's flexibility in working with a variety of public keys.

Algorithm 3.4 (a) ESS decryption - {N,X} as public key

```
Algorithm Revised ESS_Decrypt for Dual-Key Cryptosystem
1. ESS_Decrypt_X()
  Input: Ciphertext message C
  Output: Decrypted plaintext message m
  Procedure:
    1. Download the ciphertext message C from the CSP premises.
    2. Decrypt the message using private key d and modulus X:
      m = C^d mod X

2. ESS_Decrypt_Y()
  Input: Ciphertext message C
  Output: Decrypted plaintext message m
  Procedure:
    1. Download the ciphertext message C from the CSP premises.
    2. Decrypt the message using private key d and modulus Y:
      m = C^d mod Y
```

Procedure:

The Single Owner (SO) retrieves the ciphertext message, C, from the Cloud Service Provider's (CSP) location. Subsequently, using the private key d and the Modulus of Y, the SO decrypts the message, resulting in the retrieval of the plaintext message m.

Plaintext m = Cd mod Y.

2. LITERATURE REVIEW

Rui Jiang and Zhongma Zhu (2016) [13]created a safe data strategy for dynamic members. The distribution of keys was done securely by avoiding the communication routes. The plan was protected against a collusion assault in which the banned users combined with an unreliable cloud provider and were unable to obtain the original data file. When a user joined or departed a group, the planned scheme did not update the private keys. But secure data sharing required a lot of RAM.

Chavhan Bhaurao and Deshmukh Swati (2015) [14] introduced a safe multi-proprietor data sharing arrangement for cloud-based unique gatherings. The gathering signature and dynamic transmission encryption methods were utilized by the cloud client to trade the data. The processing and stockpiling above were not at the necessary level. In any case, utilizing the protected multi-proprietor data sharing strategy didn't bring down the capacity above under a specific point.

Secure data sharing was described by Mazhar Ali et al. (2015)[15] as upgrading data confidentiality, integrity, access control, sharing, and danger security. Data was encoded utilizing a solitary encryption key utilizing the SeDaSC approach. Utilizing client share, two unmistakable key offers were created for every client. the responsibility for single, vital SeDaSC strategy share for foiling insider dangers. A cryptography server, a dependable outsider, held an additional key offer. Nonetheless, utilizing Secure Data Partaking in Clouds didn't address the issue.

According to Bojan Suzic et al. (2015),[16] the private cloud federation's infrastructure enabled safe data processing and exchange. Study was done on the characteristics of data, security policy, and access control languages. Fine-grained security and data processing in semi-trusted environments were made possible by the characteristics. The primary obstacles were presented as means of facilitating interoperability amongst diverse infrastructures and services. Enabling security and legal compliance with privacy and openness for public sector needs was the main goal.

A property based data sharing method was created by Shulan Wang et al. (2016) [17] to beat the key escrow issue in CC applications. An enhanced strategy for giving two-party keys ensured that the vital authority and CSP teamed up with the client's mystery key. The weighted attribute concepts improved attribute expression and expanded it from a binary to an arbitrary state. Nevertheless, the attribute-based data sharing strategy did not result in an improvement in data secrecy.

Qinlong Huang et al. (2015) [18] demonstrated a characteristic based secure data imparting way to deal with productive disavowal (EABDS) in CC. The proposed strategy scrambled the data along with the Data Encryption Key (DEK) utilizing symmetric encryption in view of CP-ABE to save data confidentiality and achieve fine-grained access control. With the guide of trait authority and key server support, the created approach made client characteristic mystery keys utilizing homomorphic encryption. The planned methodology disposed of the trait authority from data access by making quality mystery keys. Both forward and backward security were achieved with an instantaneous attribute revocation mechanism. But employing an attribute-based secure data exchange system did not reduce the time required for key generation.

A Secure Multi-Owner Data Sharing Scheme was intended for a dynamic cloud group by Marimuthu et al. (2014)[19]. The cloud user sends the data to other users via broadcast encryption and group signing. The number of revoked users has no bearing on the scheme's computation costs or storage overhead. The access was secured by an authentication procedure called One-Time Password. Installing a secure form of authentication on several machines was possible using One-Time Password. Nevertheless, there was no secure method used for the data sharing.

Cong Wang et al. (2011) [20] improve the secure searchable index by statistical methods and preserve sensitive score information through one-to-many order-preserving mapping methods.

An Effective and Mysterious Data Sharing Convention (EFADS) with adaptable sharing was made by Guiyi Wei et al. (2014) [21] for cloud data rethinking. EFADS further developed data confidentiality and anonymity for data sharers without requiring a side to be completely trusted. This kind of intermediary re-encryption was unknown, matching free, self-deciding, and unidirectional. Nevertheless, utilizing a property based secure data trade methodology didn't bring about a more limited encryption time.

To forestall the arrival of unlawful data when a canceled client re-joined the framework, Bharath K. Samanthula et al. (2015) [22] made a compelling and Secure Data Sharing (SDS) system utilizing holomorphic encryption and intermediary re-encryption. To forestall data misfortune because of collaboration between the cloud service provider and the disavowed client, a clever arrangement was made utilizing a data conveyance approach. Vidyanand Ukey & Nitin Mishra (2014) found that data was stored on servers based on security settings, with the more secure settings giving the data priority. The data were divided into three categories in accordance with the data owner's rating of the data's relevance. Every level's data was encrypted using techniques for encryption and decryption. The main goal was to protect data storage in order to prevent attacks and invasions. The purpose of Elliptic Curve Cryptography (ECC) was cloud security. However, the security solution did not raise the level of security.

3. RESEARCH METHODOLOGY

3.1. Experimental Results

3.1.1. Performance Analysis - Variable file size

Performance evaluation of the ESS (Enhanced Schmidt Samoa) cryptosystem was performed on an owner's cloud data, analyzing its effectiveness for different input file sizes. The study compares the encryption and decryption execution times

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
***Article Received:** 10 September 2023 **Revised:** 20 October 2023 **Accepted:** 30 October 2023*

of the ESS encryption system with the traditional Schmidt Samoa (SS) encryption system, which uses a consistent key size of 32 bits.

Table 3.1 compares the durations of two different cryptographic systems in the setting of single-owner cloud data; these are the Schmidt Cryptosystem Samoa (SS) and the Enhanced Schmidt Cryptosystem Samoa (ESS). The SS system uses a simpler set of key values (p=73529) whereas the ESS system employs a more complex set (p=73529) that also includes r=73553, s=73561, N=2391448019, and d=1564100563. Cloud computing settings place a premium on data safety and privacy, therefore these cryptographic methods are essential for assuring secure data transfer and storage.

The private key (d), which is used both for encryption and decryption, is also calculated. There is a need for additional improvements to SS's robustness and scalability because its relatively simple key structure may limit its power to handle increasingly sophisticated security challenges and larger data collections.

In order to overcome some of the shortcomings of the SS system, the Enhanced Schmidt Cryptosystem Samoa (ESS) employs a more complex key configuration by adding two more prime numbers (r and s) to the already present p and q. This expansion of the key space helps to improve the encryption process's security and resilience by making it harder for adversaries to break into the system via mathematical assaults or brute force techniques. More importantly for the arena of secure cloud data management, the bigger modulus N and private key (d) in the ESS system indicate an increased capacity to handle larger datasets and increased security demands.This table includes detailed data on the duration of the encoding and decoding processes in these specific configurations, providing insight into potential performance differences between the two encoding systems in the field. Cloud data security.

Table 3.1 Time to execute SS and ESS with 32-bit key for single-owner cloud data

File size (KB)	Key Generation Time (ms)	SS for variable key size Encryption Time (us)	Decryption Time (us)	Key Generation Time (ms)	ESS for variable key size Encryption Time (us)	Decryption Time (us)
8	94	987	241	124	195	95
16		1235	425		324	190
32		2564	867		595	360
64		4659	1265		1110	715
128		6987	4567		2174	1477
256		9687	5674		4409	2990
512		12354	9874		8272	5768
1024		23564	14659		16667	11492

Because of the unpredictable activities included, the calculation cost for the vital age of ESS is higher than that of SS. Two increases, one reverse, and one lcm activity decide the calculation cost of the SS key age. To produce an ESS key, two lcm tasks, three increases, and two backwards activities are required. These decide the calculation cost. Notwithstanding, as far as encryption and decryption times, ESS outflanks SS. Since ESS involves more modest keys for encryption and decryption than SS, it has a lower computation cost.

Additionally, the Paillier, RSA, Schmidt Samoa, and Enhanced Schmidt Samoa cryptosystems have likewise been analyzed. The preliminaries were completed utilizing different document sizes in MegaBytes (MB) and a decent key size of 1024 bits. Schmidt Samoa, RSA, Paillier, and Enhanced Schmidt Samoa cryptosystems' presentation as far as encryption and decryption calculation costs are portrayed in Figures 3.1 and 3.2 for variable record estimates and fixed key sizes of 1024 bits. The results show that ESS beats the cryptosystems of Paillier, RSA, and Schmidt Samoa.

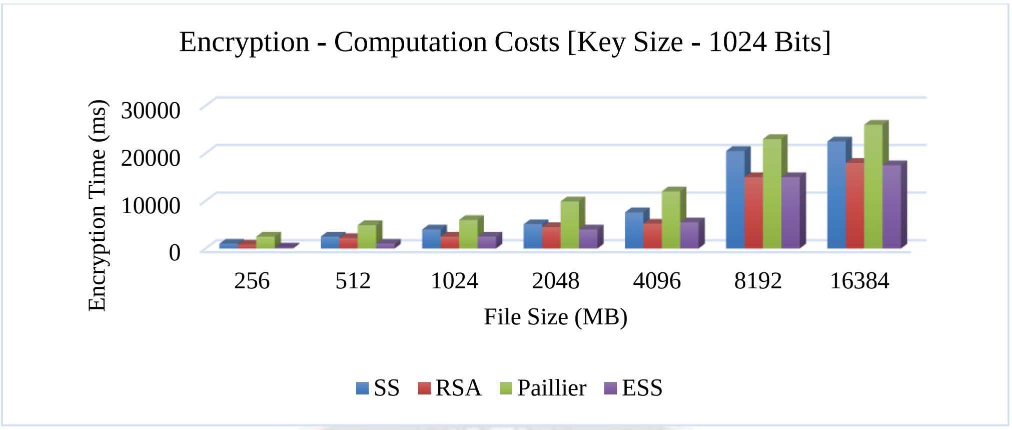

Figure 3.1 Cloud data encryption time (key length: 1024 bits).

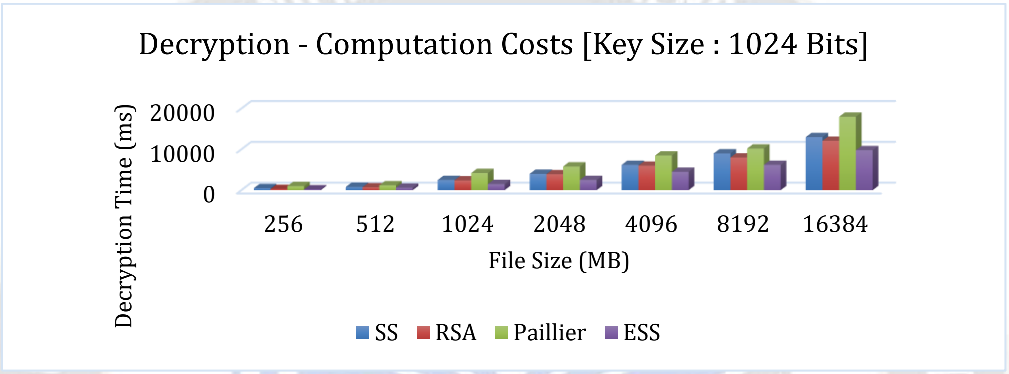

Figure 3.2 Time to decrypt data in a cloud storage system with a single owner (1024-bit key)

Fixed file size and variable key sizes have been used for the comparative study of the enhanced Schmidt Samoa, RSA, Paillier, and Schmidt Samoa cryptosystems. The performance of the RSA, Paillier, Schmidt Samoa, and Enhanced Schmidt Samoa cryptosystems is shown in Figures 3.3 and 3.4 in relation to 116 computing costs of encryption and decryption for various key sizes and fixed file sizes of 3 GB. The outcomes demonstrate that ESS outperforms the cryptosystems of Paillier, RSA, and Schmidt Samoa.

Table 3.2: Times Taken to Encrypt and Decrypt a 128-KB File at Different Key Sizes

Key size	SS Encryption (us)	SS Decryption (us)	ESS Encryption (us)	ESS Decryption (us)
4	1922	2236	1023	304
8	3078	3596	1099	406
12	6786	7546	1132	480
16	9487	9663	1107	511
24	15789	16544	1340	707
32	22478	26542	2088	1458

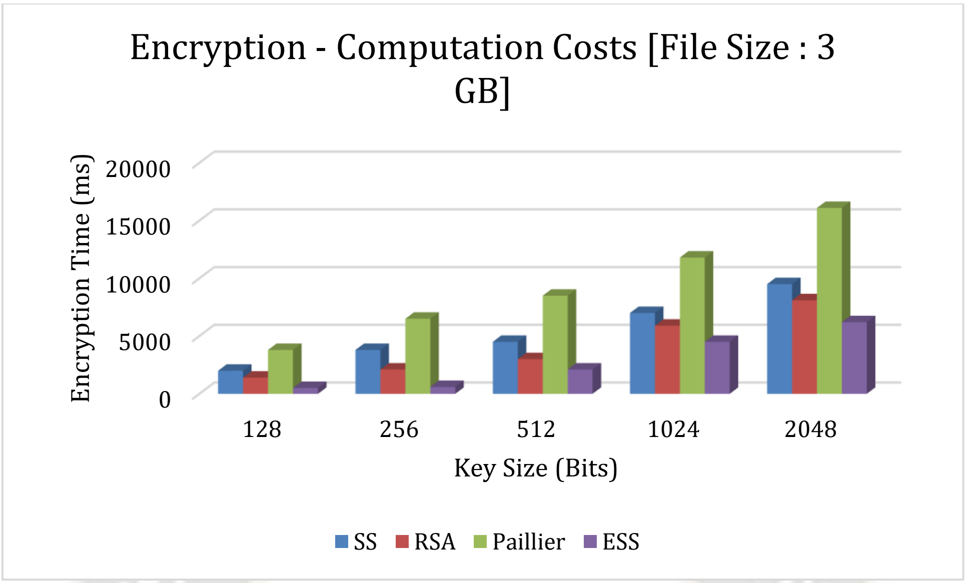

Figure 3.3: How long it takes to encrypt a 3 GB file in the cloud for a single owner.

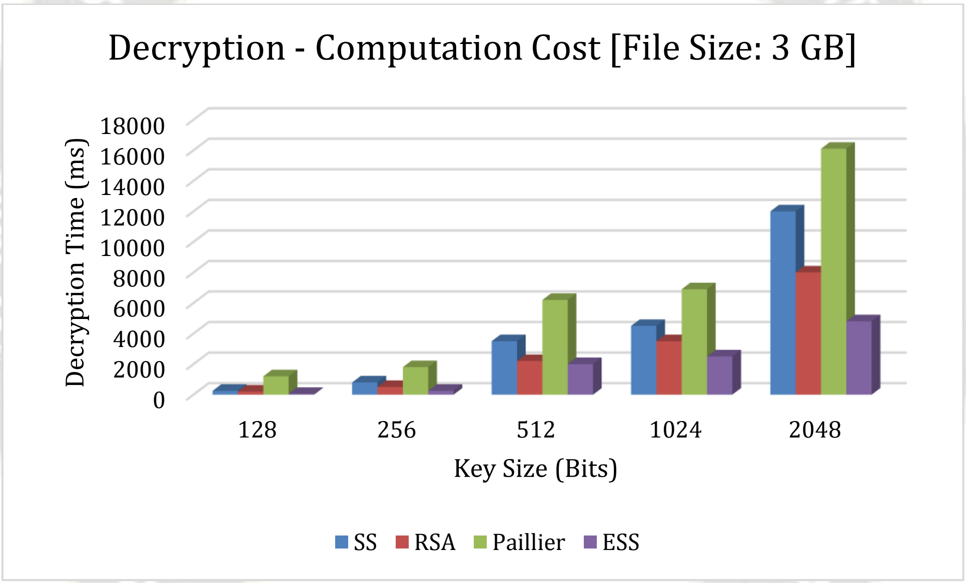

Figure 3.4: Single-owner cloud data decryption (3 GB file size)

3.2. Security Analysis

By utilizing the number field strainer technique and executing a brute power assault and whole number factorization, the security of ESS keys for single proprietor cloud data is ensured. The following parts go over the careful security examination.

3.2.1. Brute force Attack

A brute power assault is a broadly utilized classic cryptanalysis method. The objective of the assault is to interpret the given ciphertext by endeavoring each possible blend of keys. This approach gives the time expected to think twice about cryptosystem. How long it takes an interloper to break the cryptosystem concludes how secure it is. The algorithm's most significant strength is that it gives breaks additional time. The ESS takes additional opportunity to break the key than other cryptanalysis systems since it looks for every conceivable mix of the key. The vital size of the RSA, Paillier, and Schmidt-Samoa cryptosystems is exclusively subject to the two prime qualities, p and q. In any case, in the better Schmidt Samoa cryptosystem, p and q (or r and s) are the two prime qualities that decide the key size. Thus, tracking down p and q (or) r and s gives off an impression of being a difficult part of the interruption in the ESS cryptosystem.

The correlation of brute power assault times for single proprietor cloud data with various key sizes of 16, 32, 64, 128 and 256 bits and fixed plaintext document size of 512 MB is displayed in Figure 3.5. The data was gotten from the SS, Paillier, RSA, and ESS cryptosystems. Seconds (s) have been utilized to gauge the length. The chart makes it apparent that Enhanced Schmidt Samoa demands more investment to finish cryptanalysis than RSA, Paillier, and Schmidt-Samoa. Subsequently, it is trying for an assailant to break the key, and ESS has been demonstrated to be an all the more remarkable algorithm.

The SS, Paillier, RSA, and ESS cryptosystems have all been exposed to brute power assault examination utilizing an assortment of plaintext documents as info.

Figure 3.8 and the exploratory outcomes are quite similar. Coming up next is the translation of the consequences of the examination of brute power assaults: (a) The security of the 64-bit ESS key size is comparable to that of the 256-bit SS and Paillier cryptosystems; (b) The security of the 32-bit ESS key size is identical to that of the 128-bit RSA cryptosystem.

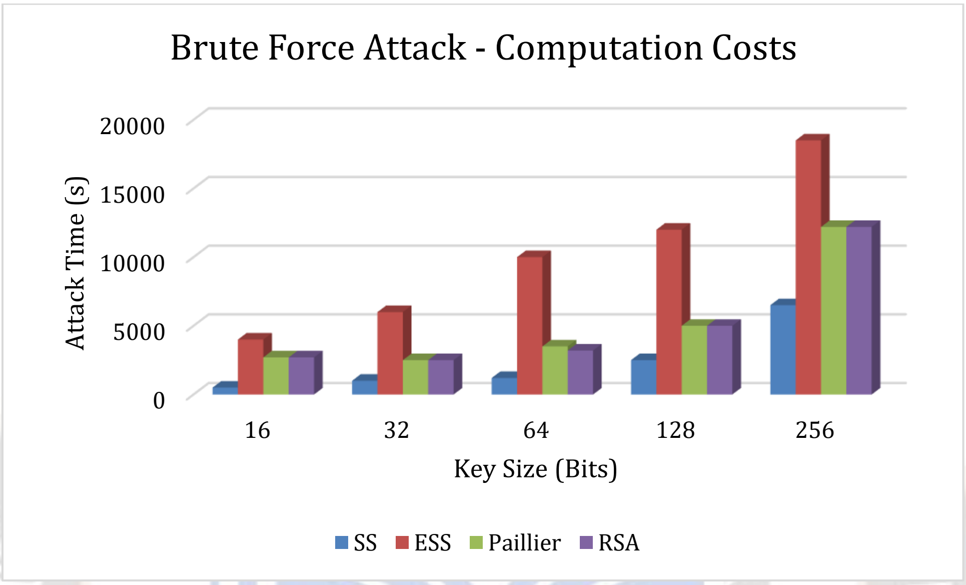

Figure 3.5: Cryptanalysis - brute force attack

3.2.2. Integer Factorization

Number factorization, or reducing a composite number to its indivisible prime elements, is a central topic in number theory. Several encryption and decryption techniques are based on this process, making it crucial in the disciplines of cryptography and security. One special case of number factorization is called prime factorization, and it entails locating the prime numbers that, when multiplied together, give rise to the original number. Finding these prime factors reveals the fundamentals of the original number, which can then be used to analyze it mathematically.

When working with huge numbers, the search for prime factors, also known as the great elements of the number, presents a considerable computational difficulty. In 2014, Shah Muhammad Hamdi et al. presented the number field sieve (NFS) method, a potent algorithm used for factoring big integers, particularly those with more than 100 digits. To ensure the safety of today's communication networks, which rely on intricate encryption methods, this technique has had a profound impact on both number theory and cryptography.

The number field sieve algorithm shows improved efficiency and scalability when applied to changeable private key sizes between 4 and 32 bits, in compared to other techniques such as the Schmidt Samoa and Enhanced Schmidt Samoa methods. For the purpose of maintaining the security and reliability of cryptographic systems, its capacity to efficiently process enormous numbers makes it an indispensable instrument. In the context of digital security, where the secrecy and privacy of sensitive data significantly depend on the efficacy of encryption techniques, this flexibility is of paramount importance.

The number field sieve method has greatly improved our grasp of number theory and its practical applications by making it possible to find prime factors in complicated and huge numbers. Its application in contemporary cryptography has been crucial in creating trustworthy channels of communication and safeguarding private data in our increasingly linked digital environment.

Table 3.3 Number field sieve cryptoanalysis for SS for single owner cloud data

Key Size	N	p	q	Attack Time (ns)
4	100	4	12	340786
8	5546	18	18	393348
12	249599	60	72	368313
16	2328161 2	260	354	381566
24	2.49E+12	13634	13526	405138
32	4.01E+14	74264	73683	595010

Table 3.4 Number field sieve cryptoanalysis for ESS for cloud data belonging to a single owner

Key Size	X	Y	p	q	r	s	Attack Time (ns)
4	326	859	16	19	24	38	742672
8	442	908	18	23	28	32	731689
12	10507	11780	102	104	108	110	722967
16	140449	157545	371	383	393	405	795202

For each key size, the table lists its corresponding coordinates (X and Y), prime numbers (p, q, and r), S value, and attack time (in nanoseconds). Sizes 4, 8, 12, and 16 are all significant here, and each has its own unique set of factors. The coordinate numbers X and Y appear to be integers that are indicative of certain features of the cryptographic procedure. Prime numbers, such as p, q, and r, are likely employed in a wide variety of computer operations and are also integral components of cryptographic techniques. The cryptographic calculations may also be affected by S, another system parameter. Another indicator of the system's resistance to potential security breaches is the time it takes for an attack to occur, which is measured in nanoseconds; bigger values indicate greater resistance to attacks.

4. CONCLUSION

A novel and efficient public key cryptosystem - Enhanced Schmidt Samoa cryptosystem has been proposed for protecting the single owner data in the cloud. Data storage will be done in the cloud once, and most of the time, data retrieval will be performed. The experimental results proved that the ESS cryptosystem could be utilized to ensure data confidentiality in the cloud. From the cloud setup and experimental results, the proposed ESS cryptosystem is highly secured and not easily breakable, compared to the traditional cryptosystems [23].

In this research, a secure cloud framework has been developed for the individuals, who are accessing the applications and data in the cloud. Individuals may generate and share the data among multiple users with reading/writing access. In organizations, mostly data will be shared among multiple users for supporting the business process. In these scenarios, multi-user (i.e.) data owner and data user, the concept needs to be adapted in the framework. If the data need to be shared, cryptographic keys need to be shared among the owners securely [24]. Improved Secure Cloud Data Storage Framework (ISCDSF) framework needs to be enhanced by addressing key management issues and ensuring the data confidentiality for multi-user data in the cloud.

REFERENCE

1. Heap-Yih Chong, et.al , An explanatory case study on Cloud computing applications in the built environment, Automation in Construction, 44 (2014) 152-162.
2. Maricela-Georgiana Avram (Olaru) , Advantages and challenges of adopting cloud computing from an enterprise perspective, Procedia technology,12 (2014) 529-534.
3. R. Velumadhava Rao, et.al, Data security challenges and its solution in cloud computing, International Conference on Intelligent Computing, Communication & Convergence (ICCC), 2015, pp. 204-209.
4. Shakeeba S. Khan, et.al, Security in Cloud Computing using Cryptographic Algorithms, International Journal of Innovative Research in Computing and Communication Engineering, 3 (2015) 148-154.
5. Rachna Arora, et.al, Secure User Data in Cloud Computing Using Encryption Algorithm, International Journal of Engineering Research and Applications (IJERA), 3 (2013) 1922-1926.
6. Randeep Kaur, et.al, Analysis of security algorithm in cloud computing, International Journal of Application or Innovation in Engineering & Management (IJAIEM), 3 (2014) 171-176.
7. Manpreet Kaur, et.al, Implementing encryption algorithms to enhance data security in cloud computing, International Journal of Computer Applications, 70 (2013) 16-21.
8. Mandeep Kaur, et.al, Using Encryption Algorithms to Enhance the Data Security in Cloud Computing, International Journal of Communication and Computer Technologies, 1 (2013) 56-59
9. Vidyanand Ukey & Nitin Mishra 2014, „Dataset Segmentation for Cloud Computing and Securing Data Using ECC‟, International Journal of Computer Science and Information Technologies (IJCSIT), vol. 5, Issue 3, pp. 4210-4213.
10. Vijay Varadharajan & Udaya Tupakula 2014, „Security as a Service Model for Cloud Environment‟, IEEE Transactions on Network and Service Management, vol. 11, Issue 1, pp. 60-75. 65.
11. Vinothkumar Muthurajan & Balaji Narayanasamy 2016, „An Elliptic Curve Based Schnorr Cloud Security Model in Distributed Environment, Hindawi Publishing Corporation, The Scientific World Journal, vol. 2016, pp. 1-8.
12. Wenfeng Wang, Peiwu Li, Longzhe Han, Shuqiang Huang, Kefu Xu, Changgui Yu & Jine Lei 2014, „An Enhanced Erasure Code-Based Security Mechanism for Cloud Storage‟, Hindawi Publishing Corporation, Mathematical Problems in Engineering, vol. 2014, pp. 1-8.
13. Zhongma Zhu & Rui Jiang 2016, „A Secure Anti-Collusion Data Sharing Scheme for Dynamic Groups in the Cloud‟, IEEE Transactions on Parallel and Distributed Systems, vol. 27, Issue 1, pp. 40-50.
14. Chavhan Bhaurao & Deshmukh Swati 2015, „Privacy Preservation and Secure Data Sharing in Cloud Storage‟, International Research Journal of Science and Engineering, vol. 3, Issue 6, pp. 231-236.

15. Mazhar Ali, Samee U Khan & Athanasios V Vasilakos 2015, „Security in cloud computing: Opportunities and challenges‟, Elsevier, Information Sciences, vol. 305, no. 1, pp. 357-383.
16. Bojan Suzic, Andreas Reiter, Florian Reimair, Daniele Venturi & Baldur Kubo 2015, „Secure Data Sharing and Processing in Heterogeneous Clouds‟, Elsevier, Procedia Computer Science, vol. 68, pp. 116-126.
17. Shulan Wang, Kaitai Liang, Joseph K. Liu, Jianyong Chen, Jianping Yu & Weixin Xie 2016, „Attribute-Based Data Sharing Scheme Revisited in Cloud Computing‟, IEEE Transactions on Information Forensics and Security, vol. 11, Issue 8, pp. 1661-1673.
18. Qinlong Huang, Zhaofeng Ma, Yixian Yang, Jingyi Fu & Xinxin Niu 2015, „EABDS: Attribute-Based Secure Data Sharing with Efficient Revocation in Cloud Computing‟, Chinese Journal of Electronics, vol. 24, Issue 4, pp. 862-868.
19. Qinlong Huang, Yixian Yang & Mansuo Shen 2017, „Secure and efficient data collaboration with hierarchical attribute-based encryption in cloud computing‟, Future Generation Computer Systems, Elsevier, vol. 72, Issue C, pp. 239-249.
20. Marimuthu, K, Ganesh Gopal, D, Sashi Kanth, K, Srujay Setty & Kunal Tainwala 2014, „Scalable and Secure Data Sharing for Dynamic Groups in Cloud‟, IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), pp. 1697-1701.
21. Cong Wang, Ning Cao, Kui Ren & Wenjing Lou 2012, „Enabling Secure and Efficient Ranked Keyword Search over Outsourced Cloud Data‟, IEEE Transactions on Parallel and Distributed Systems, vol. 23, Issue 8, pp. 1467-1479.
22. Guiyi Wei, Rongxing Lu & Jun Shao 2014, „EFADS: Efficient, flexible and anonymous data sharing protocol for cloud computing with proxy reencryption‟, Journal of Computer and System Sciences, Elsevier, vol. 80, Issue 8, pp. 1549-1562.
23. Bharath K. Samanthula, Yousef Elmehdwi, Gerry Howser & Sanjay Madria 2015, „A secure data sharing and query processing framework via federation of cloud computing‟, Information Systems, Elsevier, vol. 48, pp. 196-212.
24. Muhammad Baqer, et.al , Next Generation of Computing through cloud computing technology. 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), 2012, pp. 1-6.