International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Enhanced Sink Node Method for Protecting Data of Single Owner

Akheel Mohammed[1], Sameera Khanam Shaik[2], Ayesha[3]

[1]Professor, Department of Computer science and engineering, Dr.VRK women's College of engineering and technology, Hyderabad, India.
[2]Student (M.Tech,) Department of Computer science and engineering, Dr.VRK women's College of engineering and technology, Hyderabad, India.
[3]Assistant Professor Department of AI&DS, VJIT, Hyderabad, India

Abstract
Cloud computing services offer computing, storage, networking, and software. Users can access data from any place and any time, under the control of a cloud service provider, by using cloud storage services. CSPs face challenges in protecting their data in the cloud. Public key cryptography can provide security, confidentiality, and integrity. The Schmidt-Samoa cryptosystem (SSC) is a public key cryptosystem that is based on integer factorization hardness. When applying ENHANCED SINK NODE in the cloud, the cryptosystem takes more time to encrypt and decrypt data. In this research, we propose an enhanced Schmidt-Samoa (SSC) public key cryptosystem to address this problem. The ENHANCED SINK NODE cryptosystem uses four prime numbers, which greatly increases the difficulty of breaking the cryptosystem compared to SSC. Further testing showed that the time needed to perform data encryption and decryption in ENHANCED SINK NODE is shorter, but the time needed to do cryptanalysis in ENHANCED SINK NODE is longer than in SSC. The ENHANCED SINK NODE cryptosystem is very secure and makes data more private.

1.1 INTRODUCTION

Cloud computing stores a lot of information and can be accessed from far away by IT companies and people who use it. Nowadays, cloud storage is essential for document archives, backup, and sharing among users. Single-owner context appears in most of the personal and business scenarios. Personal identity, private documents, images, and videos will not be shared with any other user. Sensitive data, secret business documents, and personnel details will not be shared with any personnel. Maintaining the secrecy of the data is the most important feature in a single-owner context. In the above examples, only one owner is responsible for storing, manipulating, and retrieving the data. If single-owner data is outsourced to cloud storage, then it is mandatory to ensure data secrecy.

Cloud computing is a paradigm that enables on-demand access to shared pools of computing resources, such as servers, storage, networks, applications, and services. Cloud computing offers many benefits for users and organizations, such as scalability, cost-efficiency, reliability, and convenience. However, cloud computing also poses significant challenges for data security and privacy, as users must entrust their sensitive data to third-party cloud service providers (CSPs) who may not be fully trustworthy or reliable. Moreover, cloud data may be subject to various attacks from malicious insiders, hackers, or eavesdroppers who can compromise the confidentiality, integrity, and availability of the data. Therefore, it is essential to

design and implement effective cryptographic schemes that can protect cloud data from unauthorized access and manipulation.

One of the popular cryptographic techniques for securing cloud data is public-key encryption (PKE), which allows users to encrypt and decrypt data using different keys: a public key that can be shared with anyone, and a private key that is kept secret by the owner. PKE enables secure data transmission and storage without requiring prior key exchange or trust establishment between the parties. However, most existing PKE schemes are based on number-theoretic assumptions, such as the hardness of factoring large integers or computing discrete logarithms, which are vulnerable to quantum attacks. Quantum computers can potentially solve these problems in polynomial time using algorithms such as Shor's algorithm or Grover's algorithm, rendering traditional PKE schemes insecure. Therefore, there is a need for developing post-quantum PKE schemes that can resist quantum attacks and provide long-term security for cloud data.

One of the promising candidates for post-quantum PKE is the Schmidt-Samoa cryptosystem (SS), which was proposed by Claus-Peter Schnorr and Rainer Steinwandt in 2003. SS is based on the hardness of finding modular roots of large composite numbers, which is believed to be resistant to both classical and quantum attacks. SS has several advantages over other post-quantum PKE schemes, such as low computational complexity, short key size, and high encryption speed. However, SS also has some drawbacks, such as low decryption speed, susceptibility to chosen-ciphertext attacks (CCA), and lack of semantic security. Therefore, several variants and improvements of SS have been proposed in the literature to address these issues and enhance its performance and security.

In this paper, we propose an improved variant of the Schmidt-Samoa cryptosystem (SSC) for securing single owner data in the cloud. ENHANCED SINK NODE improves the decryption speed of SS by using a novel technique of partial decryption. ENHANCED SINK NODE also provides semantic security and CCA resistance by using a hash function and a random padding scheme. ENHANCED SINK NODE is suitable for securing single owner data, which means that only one user owns and controls the data and the corresponding encryption keys. This is a common scenario in cloud computing, where users> store their personal or professional data in the cloud and access it from multiple devices. ENHANCED SINK NODE ensures that only the legitimate owner can decrypt the data, and no one else, including the CSP or other users, can access or modify the data without the owner's permission.

Nevertheless, the primary concern in cloud storage is that a single owner, i.e. an individual, will not possess administrative authority over the data. Instead, the company that provides cloud services will have control over the data.

This scenario shows how data security issues like unauthorized data access and loss of data confidentiality can happen. Personal and business documents need to be protected by a secure framework. If an appropriate framework is implemented, the responsibility for data security will shift from a single owner to a cloud service provider.

Confidentiality is one of the major key issues. Single owner personal information and documents are sensitive where privacy needs to be preserved. Once an intruder acquires information regarding sensitive data, various challenges arise. Although encoding all this information is a straightforward process, it can be a thorny task, as most present requests rely on unencoded data.

The security and confidentiality of cloud data storage can be ensured by employing a cryptographic encryption technique. Cryptography is broadly divided into two categories: shared key cryptography and public key cryptography. In shared key cryptography, encryption and decryption are performed with the same key. Several prominent cloud service providers (CSPs) such as Google, Amazon, and Microsoft

employ the industry-standard Advanced Encryption Standard (AES) with a key length of 256 bits to safeguard data against unauthorised access. AES is a type of cryptosystem that uses the same secret key to both encrypt and decrypt data. CSP will perform the encryption and decryption process. If someone breaks into the CSP, they can get the passwords and keys to access the cloud data.

The public key cryptosystem is used to protect administrative privileges of files owned by individuals. Public key cryptography is a cryptographic algorithm, which uses various number theory concepts like integer factorization, discrete logarithms, etc. Two different keys, namely public and private, are used for the following two scenarios.

(I) The communication between the parties involved.,

(ii) **data storage and retrieval in the cloud**. The public key is known to both the owner and the cloud service provider, but the private key is only known to the owner. For the intruders, deriving the private key with the public key is computationally impossible with the public key.

Data encryption in the cloud is done with the public key, and data decryption is done by the single owner with the private key. This is called an asymmetric cryptosystem because it uses two different keys. Some of the common public key cryptography algorithms are RSA (Rivest, Shamir Adleman), ElGamal cryptosystem, SS (Schmidt-Samoa), ECC (Elliptic Curve Cryptosystem), Diffie Hellman key exchange, and others. These cryptosystems are also used in internet standards like Transport Layer Security (TLS), Good Privacy (PGP), GNU Privacy Guard (GPG), and Secure/Multipurpose Internet Mail Extensions (S/MIME). The Enhanced Schmidt Samoa (SSC) public key cryptosystem that we propose in the cloud gives the single owner (SO) control over their own files. The cloud service provider does other tasks like encryption, decryption, and key management, besides the existing ones. But data security is not assured if the cloud service provider is malicious. We suggest that SO chooses the key length and does encryption/decryption. If someone tries to hack the data, they won't see the real data because it's encrypted. Data confidentiality for the single owner source can be ensured by using a strong encryption algorithm and keeping the private key secret.

The contribution of this research work is to propose the Enhanced Schmidt Samoa (SSC) cryptosystem for the single owner data storage in the cloud, by optimizing the encryption time and decryption time, and by increasing the complexity of cryptanalysis. Public and private keys are made based on what needs to be done, which makes it harder to use brute force.

attack in the proposed ENHANCED SINK NODE scheme. As the security strength of ENHANCED SINK NODE surpasses that of SS, the single owner cloud framework utilizing ENHANCED SINK NODE for data storage will be more secure.1

1.2 USING SINGLE OWNER CLOUD DATA

The MIDT stands for the SAMOA CRYPTOSYSTEM

This public-key cryptosystem involves plain text, ciphertext, a public key, a private key, an encryption algorithm, and a decryption algorithm. The main uses of public key cryptography are confidentiality, digital signatures, and key exchange. The Schmidt-Samoa (SS) system is widely used because it relies on public keys.The Schmidt Samoa algorithm (Katja Schmidt Samoa 2006) for ensuring cloud data confidentiality is shown in algorithm 1.1. This algorithm works as follows: two random prime numbers, p, and q, are generated. The Rabin and RSA algorithms, where decryption is done faster, are like this algorithm. Public keys are created by squaring p and multiplying it by q. In this cryptosystem, the encryption process is slow since the sender must do a full calculation for exponentiation. The encryption

uses the public key. The encrypted text from the sender is decrypted using the Chinese remainder theorem, where the private key is used to decrypt the encrypted text.

Algorithm 1.1 Schmidt Samoa Cryptosystem for single owner cloud data

Skeen()
Input: Two large prime numbers p and q.
Output: Public key: {N}, Private key: {d}

Procedure:
/* Compute public key N */
N = p2 * q
/* Compute private key d */
L = lcm ((p-1), (q-1)), d = N-1 mod L.
Seacrest()
Input: Plaintext message m, where m < N.
Output: Ciphertext message, C
Procedure:
SO encrypts the message using public key N to obtain the ciphertext C and transfer it to CSP.
Ciphertext C = MN mod N
SS_Decrypt()
Input: Ciphertext message, C
Output: Decrypted plaintext message m.
Procedure:
SO downloads the ciphertext message, C from CSP premises.
Then, SO decrypts the message, C using private key d to obtain the plaintext.
m.
Plaintext m = Cd mod pq.

The sample problem is solved as an example using the Schmidt Samoa cryptosystem.

Generate two large prime numbers, p=7 and q=11. (Small prime numbers have taken to solve for simplicity purpose)
Compute N = p2 * q. N = 72 * 11 = 53

Compute d = N- 1 mod lcm ((p-1), (q-1)).
d = (539)-1 mod lcm ((7-1), (11-1)) = 29.
The public key N is 539, and the private key d is 29.
Let us assume plaintext m = 32.
For encryption, ciphertext C = mN mod N
C = 32539 mod 539 = 373.
For decryption, plaintext m = Cd mod pq.
m = 37329 mod (7*11) = 32.

1.3. Single owner cloud data is secured using P.

PROPOSED ENHANCED SCHMIDT SAMOA CRYPTOSYSTEM

The proposed ENHANCED SINK NODE scheme focuses on the issues of the Schmidt Samoa cryptosyst-

tem. ENHANCED SINK NODE uses four distinct prime numbers for the generation of a public key and private key. To increase the time complexity of integer factorization and the difficulty level of brute force attacks, the ENHANCED SINK NODE scheme is proposed with four large prime numbers instead of two large prime numbers. ENHANCED SINK NODE cryptosystem includes five modules,

I) SSC_Keygen() - key generation module is used to generate Public and Private Keys

ii) SSC_Encrypt_X() - encryption module, if public key is {N,X} iii) SSC_Encrypt_Y() - encryption module, if public key is {N,Y} iv) SSC_Decrypt_X() - decryption module, if public key is {N,X} v) SSC_Decrypt_Y() - decryption module, if public key is {N,Y}

For resource access or communication in the cloud, the key needs to be generated once and utilized for encryption or decryption as many times as possible. The proposed framework for securing single owner data is shown in Figure 1.1

Single owners generate the key pairs - a public key and a private key. Single owners encrypt the data, which need to be stored in the cloud. Based on the necessity, a single owner himself/herself will decrypt the data after retrieving the ciphertext from the cloud.

Figure 1.1 Framework for protecting Single Owner (SO) data.

1.3.1 ENHANCED SINK NODE Key Generation Algorithm

process 1.2 describes the proposed ENHANCED SINK NODE key generation (SSC_Keygen()) process for single owner cloud data, which uses four huge prime numbers: p, q, r, and s. The least common multiplier has been calculated as L between (p-1, q-1) and M between (r-1, s-1). Multiply the two prime integers p, q (X) and r, s (Y). The requirement that the greatest common divisor of X, L, and Y, M, must be 1. To get the value of X', find the multiplicative inverse of X mod L. Similarly, the value Y' is calculated by determining the multiplicative inverse of Y mod M. To find the value of Z, multiply X' and Y'.

If the greatest common divisor of Z and L is one, then (I) compute N as Z mod L, and (ii) compute the multiplicative inverse of N mod L as d. The public keys are {N, X}, whereas the private key is {d}. If the greatest common divisor of Z and M is one, then (I) compute N as Z mod M, and (ii) compute the multiplicative inverse of N mod M as d. The public keys are {N,Y}, whereas the private key is {d}. Because L and M are unknown to the public**,** breaking the private key requires knowledge of the public key cryptosystem.

Algorithm 1.2 ENHANCED SINK NODE Key Generation for single owner cloud data

Sockeye()
Input: Four large prime numbers p, q, r, and s.
Output: Public key: {N,X} or {N,Y}, Private key: {d}
Procedure:
/* Compute Least Common Multiplier (LCM) between p-1 and q-1 */ L = lcm((p-1), (q-1))
/* Compute Least Common Multiplier (LCM) between r-1 and s-1 */ M = lcm((r-1), (s-1))

```
/*Perform multiplication between two prime numbers*/
X = p * q & Y = r * s
/*Compute GCD & Inverses*/
If GCD (X, L) = 1 then X' = X-1 mod L
else If GCD (Y,M) = 1 then Y' = Y-1 mod M
/*Perform multiplication between X' and Y' */
Z = X' * Y'
/*Generate public and private Key*/
If (GCD(Z,L) = = 1) then N = Z mod L, d = N-1 mod L
return {N,X,d}
elseif (GCD(Z,M) = = 1) then N = Z mod M, d = N-1 mod M
return {N,Y,d}
```

1.3.2 ENHANCED SINK NODE Encryption

Based on the public key generated by the SSC_Keygen() technique, the encryption algorithms SSC_Encrypt_X() or SSC_Encrypt_Y() will be used to encrypt the cloud single owner data. Figure 3.2 depicts the flow diagram for ENHANCED SINK NODEencryption. If the public keys produced are {N, X}, the SSC_Encrypt_X() function should be used for Enhanced Schmidt Samoa encryption. The SSC_Encrypt_X() function uses plaintext's' and public keys {N,X}.

Considered as inputs. The ciphertext 'C' is then computed as plaintext'm' multiplied by the power of public key 'X' modulo another public key 'N'. If the public keys produced are {N, Y}, then the SSC_Encrypt_Y() method should be used for Enhanced Schmidt Samoa encryption. The SSC_Encrypt_Y() function accepts plaintext'm' and public keys {N,Y} as inputs. Then, ciphertext 'C' is calculated as plaintext'm' to the power of public key 'Y' over modulus of another public key.'N'.

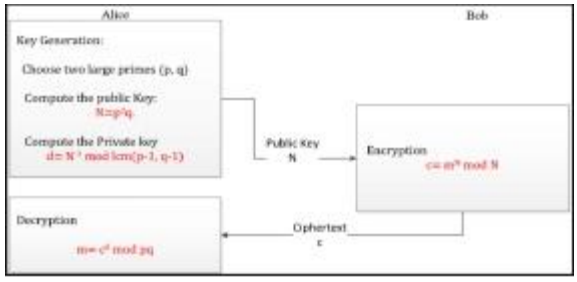

Figure 1.2 ENHANCED SINK NODE encryption for single owner cloud data

Algorithm 1.3 (a), (b) provides the pseudo-code for ENHANCED SINK NODE encryption for single owner cloud data, as shown below:

Algorithm 1.3 (a) ENHANCED SINK NODE Encryption using {N,X} as Public Key.

SSC_Encrypt_X()
Input: Plaintext message m, where m < N.
Output: Ciphertext message, C
Procedure:
SO encrypts the message using public key N, X to obtain the ciphertext C and transfers message 'C' to Cloud Service Provider (CSP) to store.
Ciphertext C = mN mod X

Algorithm 3.3 (b) ENHANCED SINK NODE Encryption - {N,Y} as Public Key

SSC_Encrypt_Y()
Input: Plaintext message m, where m < N.
Output: Ciphertext message, C
Procedure:
SO encrypts the message using public key N, Y to obtain the ciphertext C and transfers message 'C' to Cloud Service Provider (CSP) to store.
Ciphertext C = mN mod Y

1.3.3 ENHANCED SINK NODE Decryption

The decryption of the cloud single owner data will be performed using either ENHANCED SINK NODE_Decrypt_X() or ENHANCED SINK NODE_Decrypt_Y(), depending on which technique was chosen as the ENHANCED SINK NODEencryption procedure. Figure 1.3 depicts the flow of ENHANCED SINK NODEdecryption for cloud data with a single owner.

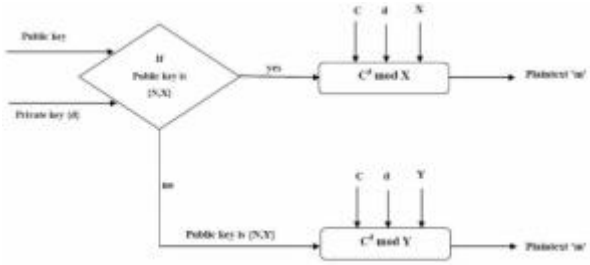

Figure 1.3 ENHANCED SINK NODE decryption for single owner cloud data

If the public key {N, X} is used for ENHANCED SINK NODE encryption, the SSC_Decrypt_X() function should be used for Enhanced Schmidt Samoa decryption. The SSC_Encrypt_X() function accepts ciphertext 'C', public key {N, X}, and private key {d} as inputs. The plaintext's' is converted into ciphertext 'C' by multiplying the private key {d} by the modulus of the public key 'X'. If the public key {N, Y} is used for ENHANCED SINK NODE encryption, then the SSC_Decrypt_Y() method must be considered for Enhanced Schmidt Samoa decryption. The SSC_Encrypt_Y() function accepts ciphertext 'C', public key {N, X}, and private key {d} as inputs. The plaintext'm' is converted to ciphertext 'C' by multiplying the private key {d} by the modulus of the public key 'Y'. The pseudo-codes of Algorithms 1.4 (a) and (b) describe ENHANCED SINK NODEdecryption for single-owner cloud data as follows:

Algorithm 1.4 (a) ENHANCED SINK NODEdecryption - {N,X} as public key

SSC_Decrypt_X()
Input: Ciphertext message, C
Output: Decrypted plaintext message m.
Procedure:
SO downloads the ciphertext message, C from CSP premises. Then, SO decrypts the message, C using private key d and Modulus of X to obtain the plaintext m.
Plaintext m = Cd mod X.

Algorithm 1.4 (b) ENHANCED SINK NODEdecryption - {N,Y} as public key

SSC_Decrypt_Y()
Input: Ciphertext message, C
Output: Decrypted plaintext message m.

Procedure:
SO downloads the ciphertext message, C from CSP premises. Then, SO decrypts the message, C using private key d and Modulus of Y to obtain the plaintext m.
Plaintext m = Cd mod Y.

1.4 ILLUSTRATION OF ENHANCED SCHMIDT SAMOA CRYPTOSYSTEM

Let's look at an example of the Enhanced Schmidt Samoa cryptosystem for single-owner cloud data. The following example applies to public key {N, X}:

ENHANCED SINK NODEKey Generation
Generate four large prime numbers, p=13, q=19, r=23, and s=37. (Small prime numbers are taken for simplicity)
Compute L = lcm((p-1), (q-1)) = lcm (12,18) = 36
Compute M = lcm((r-1), (s-1)) = lcm (22, 36) = 396
Calculate X = p * q = 13 * 19 = 247
Calculate Y = r * s = 23 * 37 = 851
Compute GCD (X,L) = GCD(247, 36) = 1.
Calculate Inverse X' = X-1 mod L = 247-1 mod 36 = 7
Compute GCD(Y,M) = GCD(851, 396) = 1
Calculate Inverse Y' = Y-1 mod M = 851-1 mod 396 = 47
Perform Multiplication Z = X' * Y' = 329
Check GCD (Z,L) = = 1, GCD(329, 36) = 1
So, calculate.
N = Z mod L = 329 mod 36 = 5
d = N-1 mod L = 5-1 mod 36 = 29
ENHANCED SINK NODE Encryption & Decryption
Let us assume plaintext m = 32.
Encryption, ciphertext C = mN mod X
C = 325 mod 247 = 223
Decryption, plaintext m = Cd mod X.
m = 22329 mod 247 = 32.

The following example describes the case of public key {N, Y}:

ENHANCED SINK NODE Key Generation
Generate four large prime numbers, p=7, q=11, r=13, and s=17. (Small prime numbers have taken to solve)
Compute L = lcm((p-1), (q-1)) = lcm (6,10) = 30
Compute M = lcm((r-1), (s-1)) = lcm (12,16) = 48
Calculate X = p * q = 7 * 11 = 77
Y = r * s = 13 * 17 = 221
Compute GCD (X,L) = GCD(77,30) = 1.
Calculate Inverse X' = X-1 mod L = 77-1 mod 30 = 23
Compute GCD(Y,M) = GCD(221, 48) = 1

Calculate Inverse Y' = Y-1 mod M = 221-1 mod 48 = 5
Perform Multiplication Z = X' * Y' = 115
Check GCD (Z,L) = = 1, GCD(115, 30) = 5
So, check GCD (Z,M) = = 1, GCD(115, 48) = 1
So, calculate.
N = Z mod M = 115 mod 48= 19
d = N-1 mod M = 19-1 mod 48 = 43
ENHANCED SINK NODE Encryption & Decryption
Let us assume plaintext m = 32.
Encryption, ciphertext C = mN mod Y,
C = 3219 mod 221 = 111
Decryption, plaintext m = Cd mod Y,
m = 11143 mod 221 = 32.

1.5 MATHEMATICAL PROOF OF ENHANCED SCHMIDT-
SAMOA CRYPTOSYSTEM
The proposed ENHANCED SINK NODE Cryptosystem for single owner cloud data is proved mathematically in the following procedure,
Case 1: {N,X} as public key
The ciphertext is determined by Equation 3.1,
C = mN mod X (3.1)
The plaintext can be computed by decrypts of the ciphertext, as given in Equation.
3.2,
m = Cd mod X (3.2)
The objective is to derive the original plaintext 'm' from Cd mod X, as given in
Equation 1.3.
Cd mod X = mNd mod X
From Algorithm 3.1,
d N-1 mod (lcm((p-1), (q-1)))
Hence,
Nd = 1 mod (lcm((p-1), (q-1)))
= 1 mod ((p-1) * (q-1) / gcd ((p-1) * (q-1)))
= 1 + k ((p-1) * (q-1) / gcd ((p-1) * (q-1))
(k is any positive integer)
(3.3)
(3.4)
Substituting Equation 1.4 in equation 1.3
Cd mod X = m1 + k ((p-1) * (q-1) / gcd ((p-1) * (q-1)) mod X
= (m * mk ((p-1) * (q-1) / gcd ((p-1) * (q-1))) mod p*q
= m * (m(p-1)) (k * (q-1) / gcd ((p-1) * (q-1)) mod p*q
By Fermat's Little theorem, Mp-1= 1 mod p
= m * (1) (k * (q-1) / gcd ((p-1) * (q-1)) mod X
= m mod X

= m (Since m<X)
Case 2: {N,Y} as public key
The ciphertext is determined by Equation 3.5,
C = mN mod Y
(1.5)
The plaintext can be computed by decrypt of the ciphertext, as given in Equation.
1.6,
m = Cd mod Y (3.6)
The objective is to derive the original plaintext 'm' from Cd mod Y, as given in
Equation 1.7.
Cd mod X = mNd mod Y (3.7)
From Algorithm 3.1,
d N-1 mod (lcm((r-1), (s-1)))
Hence,
Nd = 1 mod (lcm((r-1), (s-1)))
= 1 mod ((r-1) * (s-1) / gcd ((r-1) * (s-1)))
= 1 + k ((r-1) * (s-1) / gcd ((r-1) * (s-1)) (3.8)
(k is any positive integer)
Substituting Equation 3.8 in Equation 3.7
Cd mod X = m1 + k ((r-1) * (s-1) / gcd ((r-1) * (s-1)) mod Y
= (m * mk ((r-1) * (s-1) / gcd ((r-1) * (s-1))) mod r*s
= m * (m(r-1)) (k * (s-1) / gcd ((r-1) * (s-1)) mod r*s
By Fermat's Little theorem, Mp-1= 1 mod p
= m * (1) (k * (s-1) / gcd ((r-1) * (s-1)) mod Y
= m mod Y
= m (Since m<Y)

1.6 EXPERIMENTAL RESULTS

To simulate ENHANCED SINK NODE for single owner cloud data, we use Java Big Integer library methods (Neal R Wagner 2003) and build up a private cloud using Eucalyptus on a 2.50 GHz Intel® Core™ i5-3120M processor and 8 GB RAM. Five clusters were constructed at random, each with four nodes (owners) and mapped to a community. Each cloud node will call the SSC_Keygen() function to produce a public and private key. Then, each node will exchange its public key with the other nodes in the cluster. The proposed ENHANCED SINK NODEsystem's security and performance were analysed using this cloud arrangement. The security of SS and ENHANCED SINK NODE is assured by using a brute force attack and a number field sieve - integer factorization technique. The numerous ENHANCED SINK NODE performance metrics are examined to determine the algorithm's overall execution time.

1.6.1 Performance Analysis - Variable file size

The proposed ENHANCED SINK NODE cryptosystem’s performance for single-owner cloud data is evaluated using various input file sizes. The time required for encryption and decryption against the Schmidt Samoa cryptosystem is compared to the planned Enhanced Schmidt Samoa cryptosystem for single-owner cloud data. To compare effectively, SS and planned ENHANCED SINK NODE encryption

and decryption use tiny input file sizes ranging from 8KB to 1024 KB, with constant key sizes of 16 and 32 bits.

In SS, the key value for 16 bits is calculated as p=257, q=349, N=23051101, and d=19189. In SSC, the key value for 16 bits is chosen using p=367, q=379, r=389, s=401, N=4099, and d=8699. Table 3.1 displays the execution time of the Schmidt Samoa cryptosystem. Enhanced Schmidt Samoa cryptosystem for single-owner cloud data, with a 16-bit key.

Table 3.1: Execution time in SS and ENHANCED SINK NODE with key size 16 bits for single owner cloud data.

File size. (KB)	SS for variable key size			ENHANCED SINK NODE for variable key size		
	Key Generation Time (Ms)	Encryption Time (µs)	Decryption Time (µs)	Key Generation Time (Ms)	Encryption Time (µs)	Decryption Time (µs)
8	47	864	235	68	148	43
16		951	352		262	80
32		1023	625		476	138
64		3546	985		870	255
128		6531	1254		1113	516
256		9564	5964		3339	1019
512		10235	8546		6496	2027
1024		16547	9873		12910	4126

Again, for SS, the 32-bit key value is p=73529, q=73547, N=506233051, and d=1475417611. For SSC, the 32-bit key value is p=73529, q=73547, r=73553, s=73561, N=2391448019, and d=1564100563. Table 3.2 shows the execution times of the Schmidt Samoa cryptosystem and the Enhanced Schmidt Samoa cryptosystem for single-owner cloud data with a key size of 32 bits.

Table 3.2: Execution time in SS and ENHANCED SINK NODE with key size 32 bits for single owner cloud data.

File size. (KB)	SS for variable key size			ENHANCED SINK NODE for variable key size		
	Key Generation Time (Ms)	Encryption Time (µs)	Decryption Time (µs)	Key Generation Time (ms)	Encryption Time (µs)	Decryption Time (µs)
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

Tables 3.1 and 3.2 provide the following inferences: Because of the complicated procedures involved, the computation cost for ENHANCED SINK NODE key generation is higher than that of SS. The calculation cost for SS key generation is determined by two multiplications, one inversion, and one lcm operation. In the ENHANCED SINK NODE key generation process, the computing cost is determined by two lcm operations, three multiplications, and two inverse operations. However, in terms of encryption and decryption times, ENHANCED SINK NODE outperforms SS. ENHANCED SINK NODE has a lower calculation cost than SS due to the smaller size of the encryption and decryption keys. Furthermore, a comparison study was conducted with the Schmidt Samoa, RSA, Paillier, and Enhanced Schmidt Samoa cryptosystems. The studies were done for a given key. Size of 1024 bits with different file sizes in Megabytes (MB). Figures 3.4 and 3.5 depict Schmidt Samoa, RSA, Paillier, and Enhanced. Schmidt Samoa cryptosystems performance in terms of computing costs of encryption and decryption for variable and constant file sizes

Key size is 1024 bits. The results demonstrate that ENHANCED SINK NODE outperforms Schmidt Samoa, RSA, andPailliercryptosystems.

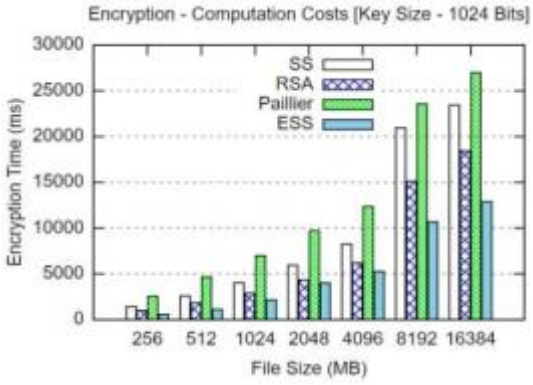

Figure 1.4: Encryption time for single owner cloud data (key size - 1024 bits).

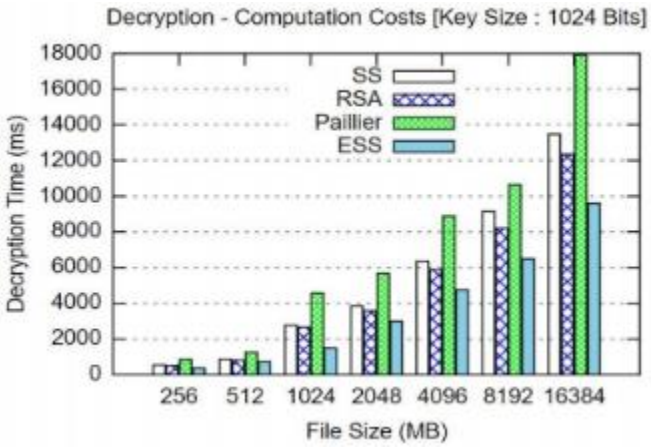

Figure 1.5: Decryption time for single owner cloud data (key size - 1024 bits).

1.6.2 Performance Analysis - Variable key size

The performance of the Enhanced Schmidt Samoa cryptosystem was further investigated using variable key sizes. Tables 3.3 and 3.4 illustrate the key generation times for the Schmidt Samoa cryptosystem and the upgraded Schmidt Samoa cryptosystem for single-owner cloud data. For analysis, the public key and private key are computed for private key sizes of 4, 8, 12, 16, 24, and 32 bits, respectively.

Table 3.3 Key generation time in SS for single owner cloud data

SS for variable key size					
Key Size	**p**	**q**	**Pub Key**	**Prv Key**	**Key Generation Time (ms)**
4	**3**	**11**	**99**	**9**	**8**
8	**17**	**19**	**5491**	**91**	**19**
12	**59**	**71**	**247151**	**1691**	**32**
16	**257**	**349**	**23051101**	**19189**	**47**
24	**13499**	**13513**	**1363151905**	**10757113**	**68**
32	**73529**	**73547**	**506233051**	**1475417611**	**94**

The findings reveal that when the private key size rises, the calculation cost of Enhanced Schmidt Samoa key generation increases in comparison to Schmidt- Samoa, owing to the complicated procedures required, as previously described. In the ENHANCED SINK NODE Cryptosystem, four huge ordered prime numbers are utilized instead of two random prime numbers, which increases the attack time. The calculation of the private key 'd' is dependent on either X or Y. Thus, the computation of 'd' is not straightforward. As a result, depending on the prime numbers, the key generation task requires a decision-making procedure, which takes more time than Schmidt-Samoa. Keys will be produced once and used as many times as feasible for encryption and decryption to provide secure communication and resource access. The higher the time it takes to generate keys increases the time it takes for an attacker to get into the ENHANCED SINK NODE cryptosystem.

Table 1.4 Key generation time in ENHANCED SINK NODE for single owner cloud data.

ENHANCED SINK NODE for variable key size							
Key Size	**p**	**q**	**r**	**s**	**Pub Key**	**Prv Key**	**Key Generation Time (ms)**
4	**17**	**19**	**23**	**37**	**133**	**13**	**12**
8	**19**	**23**	**29**	**31**	**157**	**169**	**29**
12	**101**	**103**	**107**	**109**	**2029**	**1662**	**48**
16	**367**	**379**	**389**	**401**	**4099**	**8699**	**68**
24	**9479**	**9491**	**9497**	**9511**	**11074867**	**10199203**	**98**
32	**73529**	**73547**	**73553**	**73561**	**2391448019**	**1564100563**	**124**

Table 1.5 represents the encryption time and decryption time for Schmidt.

Samoa and Enhanced Schmidt Samoa in microseconds for single-owner cloud data with a 64-KB input file size, keys in Tables 3.3 and 3.4, respectively. Similarly, Table 3.6 shows a comparison to 128 KB. In SS, encryption is done using plaintext and a public key. The public key is generated by multiplying three big prime integers. The public key is used as an exponent and modulus for the plaintext. Encryption takes longer since a huge integer was used for exponentiation.

Decryption in SS uses ciphertext and private keys. The value of 'L' is calculated by taking the least common multiple of two huge integers. The private key was calculated by taking the modular inverse of the public key, using L as the modulus. The private key was used to exponentiate the ciphertext. The

exponentiation result is then taken modulus with the multiplication value of two huge prime integers. Compared to encryption, the SS cryptosystem decrypts quicker.

The public key for ENHANCED SINK NODE key creation using four big prime integers is either {N, X} or {N, Y}. The values of N, X, and Y are created using a least common multiplier, modular inverse, or multiplication operations, which is a small quantity when compared to the SS algorithm's public key. The public key 'N' has been used to exponentiate plaintext. The modulus for the exponentiation result is then calculated using the public key 'X' or 'Y'. ENHANCED SINK NODE encrypts quicker than SS because the public key values are smaller.

Decryption in ENHANCED SINK NODE uses the public key 'X' or 'Y', the private key 'd', and the ciphertext. The value of 'L' was calculated by finding the least common multiple of two huge integers. The value of 'M' was calculated by finding the least common multiple of another two huge integers. The private key 'd' was generated by taking the modular inverse of the public key **'N' over 'L' or 'M'. The private key 'd' is used to exponentiate the ciphertext. The exponentiation result is then taken modulus with the public key 'X' or 'Y'. ENHANCED SINK NODE decrypts quicker than SS because the private key 'd' and public key 'X' or 'Y' have smaller values.**

SSC's public and private key sizes are less than those of SS. As a result of the key size utilized in encryption and decryption techniques, ENHANCED SINK NODE has a lower computational cost than SS. According to this performance analysis, when the private key size increases, the ENHANCED SINK NODE encryption and decryption execution time will be less than the SS encryption and decryption execution time for single-owner cloud data.

The Schmidt Samoa, RSA, Paillier, and Enhanced Schmidt Samoa cryptosystems were then compared using a fixed file size and varying key sizes. Figures 3.6 and 3.7 demonstrate the performance of the Schmidt Samoa, RSA, Paillier, and Enhanced Schmidt Samoa cryptosystems in relation to

The cost of encryption and decryption with varying key sizes and a fixed file size of 3 GB. The results demonstrate that ENHANCED SINK NODE outperforms Schmidt Samoa, RSA, and Paillier cryptosystems.

Table 1.5 shows the encryption and decryption times of SS and ENHANCED SINK NODE for single-owner cloud data with a file size of 64 KB.

Key size	SS for variable key size and file size 64 KB		ENHANCED SINK NODE for variable key size and file size 64 KB	
	Encryption Time (µs)	Decryption Time (µs)	Encryption Time (µs)	Decryption Time (µs)
4	1248	466	535	188
8	3549	1266	530	203
12	6906	1704	577	234
16	9011	6597	577	250
24	85645	9587	748	358
32	102256	23544	1090	732

Table 3.6 Encryption and decryption time of SS and ENHANCED SINK NODE for single owner cloud data with file size 128 KB.

Key size	SS for variable key size and file size 128 KB		ENHANCED SINK NODE for variable key size and file size 128 KB	
	Encryption Time (µs)	Decryption Time (µs)	Encryption Time (µs)	Decryption Time (µs)
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

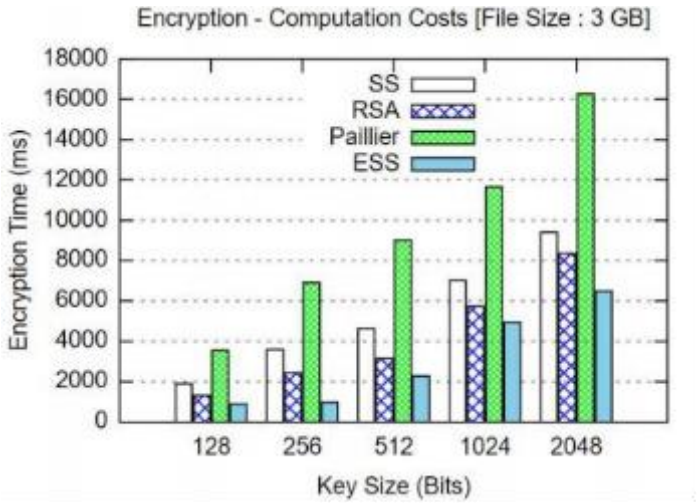

Figure 3.6 Encryption time for single owner cloud data (file size - 3 GB)

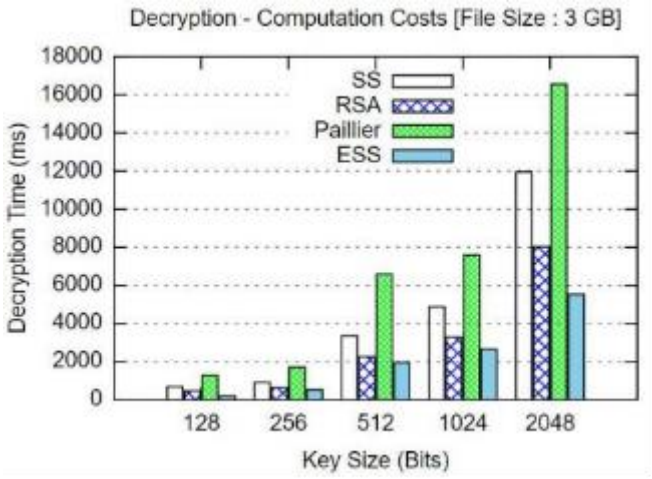

Figure 1.7 Decryption time for single owner cloud data (file size - 3 GB

1.7 SECURITY ANALYSIS

The security of ENHANCED SINK NODE keys for single owner cloud data is ensured by performing a brute force attack and integer factorization (Haibo Yu & Guoqiang Bai 2015; Arjen K, Lenstra 2000) using the number field sieve method. The detailed security analysis is discussed in the following sections.

1.7.1 BRUTE FORCE ATTACK

Brute force attacks are a common traditional cryptanalysis approach. The attack consists of attempting to determine the plaintext for the given ciphertext using all possible key sets. This attack calculates the amount of time needed to crack the cryptosystem. The time it takes an attacker to break into a cryptosystem determines its security. The algorithm's greatest strength is its increased break time. Because it tests for all potential key combinations, the ENHANCED SINK NODE takes longer to break a key than other cryptanalysis approaches. The Schmidt-Samoa, RSA, and Paillier cryptosystems only need two prime values, p, and q, to determine key size. However, with the improved Schmidt Samoa cryptosystem, the size of the key is determined by two prime numbers: p and q, or r and s. As a result, with the ENHANCED SINK NODE cryptosystem, identifying either p and q (or) r and s appears to be a challenging task for the intruder.

Figure 1.8 compares brute force attack times for SS, Parlier, RSA, and ENHANCED SINK NODE cryptosystems for single owner cloud data. Key sizes range from 16 to 256 bits, with a constant plaintext file size of 512 MB. The time taken was measured in seconds. The graph clearly shows that Enhanced Schmidt Samoa requires more time to do cryptanalysis than Schmidt-Samoa, Paillier, and RSA. As a result, the intruder finds it difficult to crack the key, and ENHANCED SINK NODE has shown to be a stronger algorithm.

The brute force attack study of SS, Parlier, RSA, and ENHANCED SINK NODE cryptosystems was carried out using various plaintext files as input.

The experimental results are fairly like Figure 1.8. The brute force attack analysis results can be interpreted as follows:

(a) The security of the ENHANCED SINK NODE cryptosystem’s 64-bit key size is equivalent to the 256-bit key size of the SS and Paillier cryptosystems.

(b) the security of the ENHANCED SINK NODE cryptosystem’s 32-bit key size is equivalent to the 128-bit key size of the RSA cryptosystem.

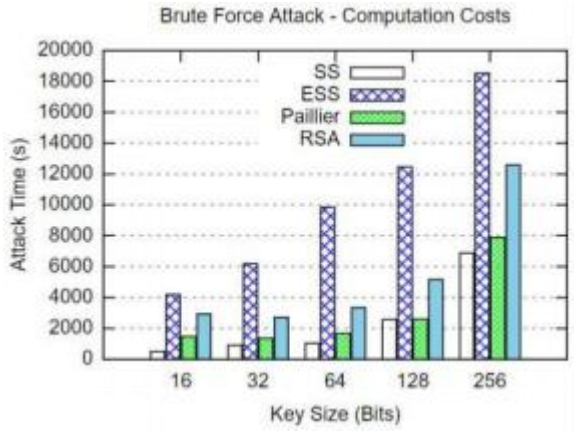

Figure1.8:CryptanalysisBruteForceAttack

1.7.2IntegerFactorization

Integer factorization, like number theory, involves breaking down a composite number into two tiny

integers. Furthermore, if these smaller integers are confined to prime numbers, the factorization becomes known as prime factorization. Factoring a positive integer ‘n’ entail finding positive prime numbers 'x' and 'y' so that the product of 'x' and 'y' will result in 'n', if x > 1 and y > 1. The prime factors of 'n' are 'x' and 'y', and the factorization of n is n = x · y. For example, n = 21 may be factored using the primes x = 7 and y = 3. In the case of huge the number field sieve technique is used to compute the prime factors of 'n'. In number theory, the universal number field sieve technique (Shah Muhammad Hamdiet al. 2014) is capable of factorizing numbers with more than 100 digits. The number field sieve technique is used to compare Schmidt Samoa and Enhanced Schmidt Samoa for varying private key sizes of 4, 8, 12, 16, 24, and 32 bits.

Table 1.7 Cryptanalysis for SS for single owner cloud data using number field sieve.

Key Size	N	p	q	Attack Time (ns)
4	99	3	11	337412
8	5491	17	19	389454
12	247151	59	71	364667
16	23051101	257	349	376798
24	2.46238E+12	13499	13513	401127
32	3.97633E+14	73529	73547	589118

Table 1.8 Cryptanalysis for ENHANCED SINK NODE for single owner cloud data using number field sieve.

Table 1.7 clearly shows the time required to execute cryptanalysis for Schmidt Samoa using the number field sieve technique for various key sizes of 4, 8, 12, 16, 24, and 32 bits. This is accomplished by factoring the value of N into three prime numbers. Table 3.8 illustrates the time required to execute cryptanalysis for Enhanced Schmidt Samoa using the number field sieve technique with key sizes of 4, 8, 12, 16, 24, and 32 bits. It is accomplished by factoring the X and Y values into two prime integers, p, and q (or r and s), respectively. Tables 3.7 and 3.8 show that Enhanced Schmidt **Samoa requires more time to perform cryptanalysis than Schmidt-Samoa does.**

1.8 SUMMARY

An innovative and efficient public key cryptosystem—Enhanced Schmidt the Samoa cryptosystem has been suggested to safeguard single-owner data in the cloud. Data will be stored in the cloud once and then retrieved most of the time. The experimental findings demonstrated that the ENHANCED SINK NODE cryptosystem might be used to maintain data secrecy in the cloud. The cloud setup and testing results show that the proposed ENHANCED SINK NODE cryptosystem is more secure and difficult to breach when compared to standard cryptosystems.

This study created a safe cloud architecture for persons who use cloud-based apps and data. Individuals can produce and share data with numerous users who have reading/writing access. In businesses, data is typically shared among various users to support business processes. In these cases, the idea of multi-user (i.e., data owner and data user) must be changed within the framework. If the data needs to be shared, the owners' cryptographic keys must be safely transferred. The Improved Secure Cloud Data Storage

Framework (ISCDSF) framework should be improved by solving critical management challenges and assuring data confidentiality for multi-user data in the cloud.