Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 1 (2025)

Securing Single Owner Cloud Data using Proposed Enhanced Schmidt-Samoa Cryptosystem

[1]Shivani Namala, [2]Dr. Rakesh Kumar Yadav

[1]Research Scholar, Department of Computer Science and Engineering, Maharishi School of Engineering and Technology, MUIT, Lucknow

shivaninamala1@gmail.com

[2]Associate Professor, Department of Computer Science and Engineering, Maharishi School of Engineering and Technology, MUIT, Lucknow

Article History:

Received: 12-11-2024

Revised: 15-12-2024

Accepted: 20-01-2025

Abstract:

In this paper an Enhanced Schmidt Samoa cryptosystem is proposed to secure single owner data storage and retrieval in a secure cloud framework. Existing and Enhanced Schmidt Samoa cryptosystem, an example of the proposed cryptosystem are discussed. This paper discusses the enrichment of the proposed Improved Secure Cloud Data Storage Framework by providing confidentiality with the enhanced Schmidt Samoa cryptosystem (ESS) to secure data storage and retrieval in cloud. Besides, an illustration of proposed cryptosystem with an example and mathematical proof of ESS are provided.

Keywords: Cloud computing, cloud security, cloud data storage, enhanced Schmidt-Samoa Cyptosystem, single owner cloud data etc.

INTRODUCTION

Cloud computing technology provides utilization of shared pool of resources such as computational power, storage facilities, networking elements, and applications as configurable services as per the cloud user's demand, without any special consideration. The cloud computing architecture holds the components and subcomponents, which is classified as front end & back end. They are connected by the Internet through a delivery system. The front end part comprises the interfaces and applications using which cloud user can access cloud. Likewise, the back end contains computing resources, namely servers, data storage, virtual machines, etc.

It enables Cloud users to access their data anytime and from anywhere, not confined to the data's storage location. Cloud computing's essential characteristics are rapid elasticity, on-demand self-service, measured service, resource pooling, multitenancy, and virtualization. Some cloud service providers are Microsoft for Azure, IBM for Bluemix, Amazon for AWS, etc. They provide services via service models like as infrastructure as aservice, platform as a service, & software as service, as well as deployment types such as public cloud, private cloud, community cloud, & hybrid cloud.

Cloud computing application areas are education, healthcare, banking, business, science and engineering design, military, transport, entertainment, governance, etc. Educational institutions have upgraded the computing systems for learner's community benefits using cloud services without the significant investment for infrastructure. The learner's world population can be easily connected and highly networked for knowledge sharing, research, and innovation activities.

Cloud computing helps the healthcare domain with ondemand computing and storage services majorly for handling electronic patient records, healthcare portals, IoT-based medical devices, and decision support systems. It also helps to connect patients and doctors remotely and from anywhere for treatment or consulting purposes.

The huge capital IT infrastructure investments are reduced to operational expenses for banking, finance, and insurance industries. The industries may focus on core functional activities instead of IT complications. Securing transactions and periodic backup or maintenance, initiating payments and fund transfers, resource planning, and customer relationship applications are hosted in the cloud for handling massive transactions.

To enhance business productivity and manage the dynamism in the business process with a massive amount of information, all business owners migrate their operations and data to cloud computing. Sales and marketing, banking, retailing, trading like business activities have been monitored through cloud computing applications

LITERATURE REVIEW

Lu Ming Xin et al. (2007) designed a symmetric key cryptosystem based on DNA biotechnology. DNA probes are used to generate keys to perform encryption & decryption. DNA fabrication and hybridization processes mapped as encryption and decryption algorithms, respectively. The proposed cryptosystem security relies on biological security and computational difficulty. However, further, the cryptosystem need to be enhanced as standard cryptosystem.

HovavShacham & BrentWaters (2008) proposed proof of retrievability in two schemes with security proofs against the untrusted server's adversaries. The first scheme is based on the random oracle model and BLS signature. Here, the client or any user can act as a verifier to check the integrity of the stored data, known as public verification. Pseudo-random functions are used in the second scheme. This is private verification, where the data owner only can verify the correctness of the stored data. Both schemes consume more time in generating and communicating client challenges to the untrusted server.

Chris Erway et al. (2009) presented provable data possession for dynamic data in outsourced file systems & version control systems. The presented scheme verifies integrity of the dynamically updated data using rank-based authenticated skiplists data structure. The challenge &response of the integrity verification were performed without retrieving original data. Experimental results show that proposed protocol is not concrete for real-time applications.

Lai Xue Jia etal. (2010) proposed asymmetric encryption & signature method based on DNA technology, where 2 pairs of keys are used for encryption & signature, respectively. Like the traditional cryptosystem, two keys are used, where the receiver public key is used for encryption & receiver's private key used for the decryption process. For signature, the sender's private key is used for signing. The sender's public key is used for signature verification. DNA molecules are used to represent keys and ciphertext. The proposed cryptosystem is efficient, secure because of DNA biological process and immune from quantum computing based attacks.

Giuseppe Atenieseetal. (2011) identified another provable data possession scheme for reducing communication cost, computation cost, and block access cost compared to previous schemes. The research proposes two provable data possession schemes, namely S-PDP & E-PDP. S-PDP scheme proves the data possession at the untrusted server. E-PDP improves the performance of SPDP by reducing the computation on the server & client-side. The proposed scheme ensures the integrity of the outsourced data in the untrusted server.

Peng Yong et al. (2012) analyzed the researcher's contributions to adapt cryptographic techniques for cloud storage security. The link between cloud storage and&cryptographic techniques was examined in order to determine the chance of cryptographic approaches being used in cloud storage. According to the research findings, using cryptographic techniques in cloud storage protects the security of data saved in the cloud and gives a research path for creating efficient cryptographic solutions to improve cloud storage security.

Cong Wangetal. (2013) suggested a privacy-preserving public auditing method to ensure the integrity of cloud data. The study focuses on data auditing utilizing TPA with the least amount of communication and processing overhead. The protocol employs random masking & homomorphic linear authenticator to protect data privacy (i.e., TPA cannot discover the real contents stored in the cloud while doing audits). Message authentication code and cryptographic hash functions are used to perform public auditing. The implementation of the protocol for real-time applications in the public cloud is left as future extension work.

Yushu Zhangetal. (2014) studied the security issues of DNA coding based symmetric ciphers. Random coding and stream ciphers are considered to be general models. Chaos theory and DNA computing are used to develop encryption schemes with general models. The research work provides a base to develop and design DNA based cryptosystems.

Chang Liu etal. (2015) address the issue of merging public auditing and block indices authentication to ensure availability &integrity of numerous clones of data. For undertaking public auditing, the study presents an authenticated data structure with a multi-replica Merkle tree. The proposed auditing technique confirms the integrity of selected blocks as well as all data file copies. The technique can carry out verifiable dynamic updates. The study provides recommendations for improving the auditing method for streaming and dynamic data in the cloud.

Samiha Marwan et al. (2016) combined DNA cryptography and steganography techniques to ensure data security. Modified DNA based playfair cipher used to perform DNA encryption, and then, the cipher results hidden in DNA media. The experimental results show that the proposed cipher is efficient compared to existing ciphers such as vignere, RSA, AES, & DNA based playfair ciphers. The proposed cipher suggested being furtherly enhanced by incorporating the biological process of DNA.

Su Peng et al. (2017) disclosures the auditors problem in identifying public keys of cloud users from public key infrastructure, which is resource consuming and complicated. The research proposes an identity-based multireplica public auditing scheme to verify replicas by TPA without public key infrastructure. The scheme supports the privacy preservation and retrievability of data. The research left dynamic updates and security issues as an open challenge.

Xing-Quan Fu etal. (2018) proposed image encryption and transmission system based on DNA encryption algorithms and double chaos systems such as coupled map lattice and optical chaotic systems. The proposed solution resists statistical, entropy, differential, and brute force attacks. The encryption, transmission, & decryption process of the proposed solution proves that system is operative and safe.

Su Peng etal. (2019) proposed a multi-replica provable data possession scheme to avoid the problem of handling public keys by TPA. The scheme utilizes a compressed authentication array and homomorphic verification tag to perform multi-replica auditing and dynamic updates. The research suggests improving the performance of the scheme by reducing the computation &communication overheads for cloud users, cloud servers, and TPA.

Hossein Nematzadeh et al. (2020) suggested a novel symmetric encryption technique based on a Binary Search Tree (BST) and DNA sequence. The suggested encryption technique has been subjected to entropy analysis, statistical attack, brute force assault, and differential attack analysis. The study intends to improve the cryptosystem using appropriate data formats.

Viswanath and Krishna etal. (2021) used a hybrid (AES) encryption technology using s-box &Feistal network to improve security in a multi Cloud scenario. The hybrid encryption approach framework splits the data, indexes it, and encrypts it. The hybrid encryption architecture outperforms the AES &Triple DES techniques. The hybrid encryption framework approach is resistant to DoS attacks, tampering attacks, and insider attacks. The hybrid encryption system encrypts both properties and data, but it takes longer to compute.

Ramachandra etal., (2022) investigate large data cloud application security &monitoring in order to host extremely sensitive data for Cloud platforms. To address this issue, the (TDES) technique is suggested to offer security for massive data on the Cloud. The suggested TDES approach provides considerably easier way for protecting against assaults and defending data privacy by increasing length of keys in (DES). The experimental findings demonstrated that the suggested TDES approach is successful in providing security &privacy to large amounts of healthcare data on the Cloud. When compared to the existing (IFHDS) technique, the suggested TDES methodology required less encryption and decryption time.

Cuzzocrea etal.(2022) used an attribution based method to improvedata security in Cloud. The presented solution used an appropriate encryption mechanism for the dominating relationship. However, while processing bigger databases, the SB-DS approach &the standard attribution-based technique increase calculation time.

A, Reyana etal., (2023) provides a unique approach to protect integrity of data & improve access control. To accomplish a unique improved storage retrieval technique is developed to increase performance of cloud's storage and retrieval procedures. The upload, download, encryption, and decryption times are all considered while evaluating the approach. The time it takes to upload afile rises in proportion to its size. Similarly, the time required to encrypt data of various sizes & formats demonstrated that it is dependent on file size and format. As a result, the encryption time grows as file size increases, illustrating suggested system's performance.

OBJECTIVES OF THE STUDY

- To enrich the Improved Secure Cloud Data Storage Framework by providing and ensuring major security principles
- To offer data secrecy in the Improved Secure Cloud Data Storage Framework for data of a single owner in cloud.

.SECURING SINGLE OWNER CLOUD DATA

A public-key cryptosystem consists of the following components: plain text, cypher text, a public key, a privatekey, an encryption method, & a decryption algorithm. A major application of public key cryptography is confidentiality, digital signature, and key exchange. Among the public key cryptosystems, Schmidt Samoa (SS) cryptosystem depends on the difficulty of integer factorization developed by Katja Schmidt-Samoa.

Algorithm 1 provides the Schmidt Samoa algorithm (Katja Schmidt Samoa 2006) for achieving cloud data secrecy, which may be defined as follows: Two prime numbers, p & q, are produced at random. It is comparable to the Rabin and RSA algorithms in that decryption is conducted more quickly. Taking the square of p and multiplying it by q yields public key. To construct the private key, take inverse of public key and determine modulus with the lcm of (p-1) and (q-1). Because the sender must conduct a whole computation for exponentiation, the encryption process becomes sluggish with this cryptosystem. The encryption is carried out using public key. To decode, private key is utilised.

Algorithm 1: Schmidt Samoa Cryptosystem for single owner cloud data

SS_Keygen()

Input: p & q are two huge prime numbers.

Output: N is public key, while d is the private key.

/* Compute the public key N */ N = p2 * q /* Calculate the private key d */ L = lcm ((p-1), (q-1)), & d = N-1 mod L.

SS_Encrypt()

Input: A plaintext message oflength m, where m N.

C ciphertext message as output

SO encrypts the message using public key N to acquire the ciphertext C and sends it to CSP.

C = mN mod N in ciphertext

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 plaintext m.

Plaintext m = C^d mod pq.

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

Generate two largeprime numbers, p = 7 and q = 11. (Small prime numbers have taken to solve for simplicity purpose)

Compute N = p^2 * q. N = 72 * 11 = 539.

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, & private key d is 29.

Let us assume plaintext m = 32.

For encryption, ciphertext C = m^N mod N

C = 32^{539} mod 539 = 373.

For decryption, plaintext m = C^d mod pq.

m = 373^{29} mod (7*11) = 32.

The proposed ESS scheme focuses on the issues of the Schmidt Samoa cryptosystem. ESS uses four distinct prime numbers for generation ofa public key & private key. In order to increase time complexity of integer factorization and the difficulty level of brute force attacks, the ESS scheme is proposed with four large prime numbers instead of two large prime numbers. ESS cryptosystem includes five modules,

- ESS_Keygen() - keygeneration module is used to generate Public & Private Keys.
- ESS_Encrypt_X() - encryption module, if public key is {N,X}
- ESS_Encrypt_Y() - encryption module, if public key is {N,Y}
- ESS_Decrypt_X() - decryption module, if public key is {N,X}
- ESS_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 isshown in Figure 1.

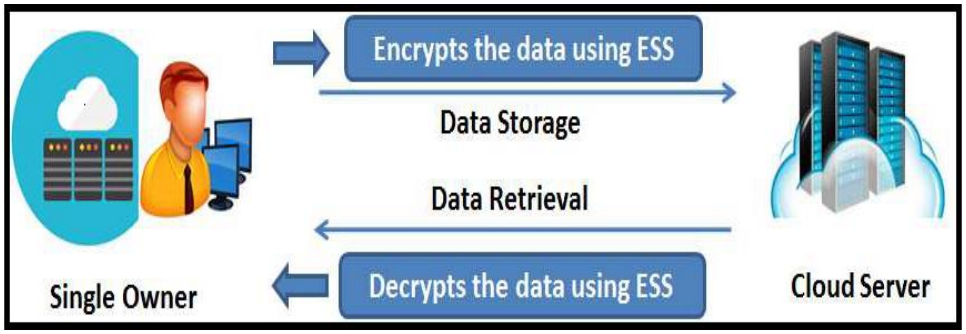

Figure 1: Proposed frameworks for securing Single Owner (SO) data

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

ESS KEYIGENERATION ALGORITHM

The proposed ESS key generation (ESS_Keygen()) algorithm for single owner cloud data is given in algorithm 2, which makes use of four largeprime numbers namely p, q, r, and s. The least common multiplier has been computed as L between (p-1), (q-1), and as M between (r-1), (s-1). Multiply the two prime numbers p, q as X and r, s as Y. The mandatory condition that greatest common divisors between X, L, and Y, M must be 1. The value of X' is computed by finding the multiplicative inverse of X mod L. Similarly, the value Y' is computed by finding multiplicative inverse of Y mod M. Multiple X' and Y' to get the value of Z.

If greatest common divisor of Z & L is one, then –

- Compute N as Z mod L
- Perform multiplicative inverse of N mod L as d.

So the public keys are {N, X} & private key is {d}. Else if the greatest commondivisor of Z and M is one, then

- Compute N as Z mod M
- Perform the multiplicative inverse ozf N mod M as d.

The public keysare {N, Y}& private key is {d}. Since L and M are not publicly known, it is difficult to break private key by knowing public key cryptosystem.

Algorithm 2: ESS Key Generation for single owner cloud data

ESS_Keygen()

Input: Four largeprime 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 $X' = X^{-1} \bmod L$

else If GCD (Y,M) = 1 then $Y' = Y^{-1} \bmod M$

/*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}

ESS ENCRYPTION

Based on the public key generation from ESS_Keygen() algorithm, encryption algorithms ESS_Encrypt_X() or ESS_Encrypt_Y() will be selected to perform encryption of the cloud single owner data. Figure 3.2 provides the flow diagram of ESS encryption. If the public keys generated are {N, X} then ESS_Encrypt_X() method needs to be considered for Enhanced Schmidt Samoa encryption. In ESS_Encrypt_X() method, plaintext 'm' and public keys {N,X} are considered as inputs. Then ciphertext 'C' is computed as plaintext 'm' to the power of public key 'X' over modulus to another public key 'N'. Else if the public keys generated are {N, Y}, then ESS_Encrypt_Y() method need to be considered for Enhanced Schmidt Samoa encryption.

In ESS_Encrypt_Y() method, plaintext 'm' and public keys {N,Y} are considered as inputs. Then, ciphertext 'C' is computed as plaintext 'm' to the power of public key 'Y' over modulus to another public key 'N'.

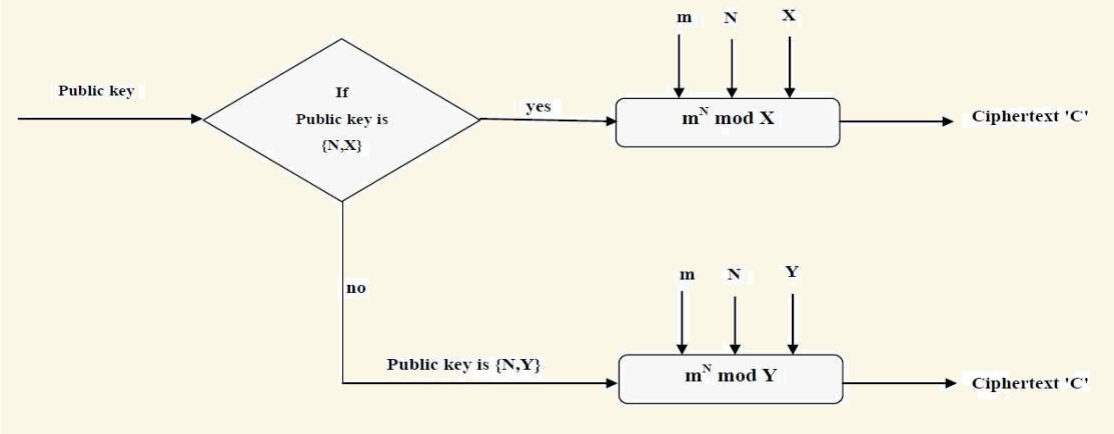

Figure 2: ESS encryption for single owner cloud data

The pseudo-code of ESS encryption for single owner cloud data is given in algorithm 3 (a), (b), as follows:

Algorithm 3 (a): ESS Encryption - {N,X} as Public Key

ESS_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 $C = m^N \bmod X$

Algorithm 3 (b): ESS Encryption - {N,Y} as Public Key

ESS_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

ESS DECRYPTION

Based on the methods ESS_Encrypt_X() or ESS_Encrypt_Y() used as ESS encryption process, either ESS_Decrypt_X() or ESS_Decrypt_Y() method will be selected to perform decryption of the cloud single owner data. Figure 3 provides the flow diagram of ESS decryption for single owner cloud data.

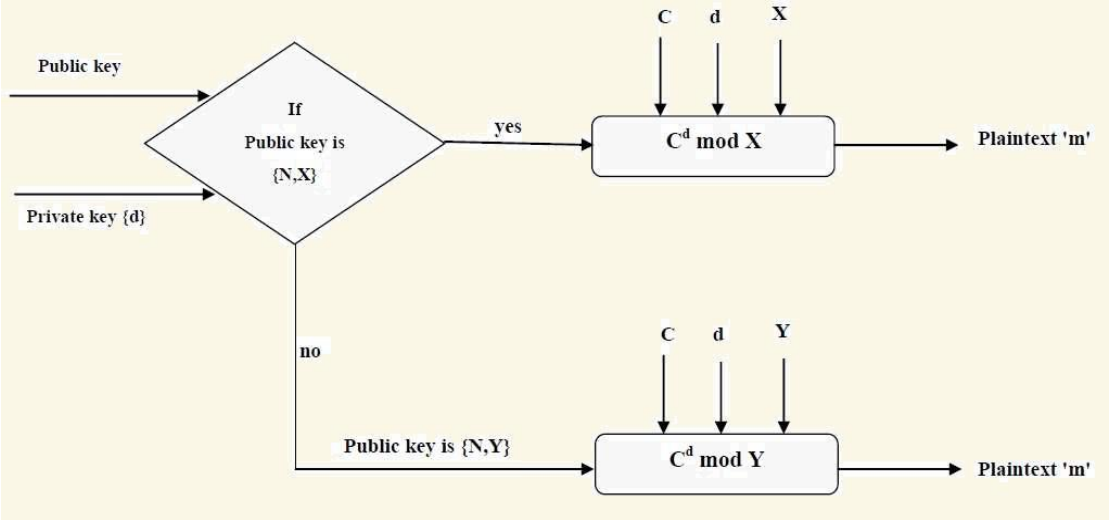

Figure 3: ESS decryption for single owner cloud data

If the public key {N, X} is used for ESS encryption, then ESS_Decrypt_X() method needs to be considered for Enhanced Schmidt Samoa decryption. In ESS_Encrypt_X() method, ciphertext 'C', public key {N, X},private key {d} are considered as inputs. Then plaintext 'm' is computed as ciphertext 'C' to the power of the private key {d} over modulus to public key 'X'. Else if the public key {N, Y} is used for ESS encryption then S_Decrypt_Y() method needs to be considered for Enhanced Schmidt Samoa decryption. In SS_Encrypt_Y() method, ciphertext 'C', public key {N, X}, private key {d} are considered as inputs. Then plaintext 'm' is computed as ciphertext 'C' to the power of the private key {d} over modulus to public key 'Y'. The pseudo-code of ESS decryption for single owner cloud data is given in Algorithm 4 (a) and (b), as follows:

Algorithm 4 (a): ESS decryption - {N,X} as public key

ESS_Decrypt_X()

Input: Ciphertext message, C

Output: Decrypted plaintext message m.

Procedure:

SO downloads 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 $m = C^d \bmod X$.

Algorithm 4 (b): ESS decryption - {N,Y} as public key

ESS_Decrypt_Y()

Input: Ciphertext message, C

Output: Decrypted plaintext message m.

Procedure:

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

Plaintext $m = C^d \bmod Y$.

ILLUSTRATION OF ESSC

Let us discuss an example for Enhanced Schmidt Samoa cryptosystem for single owner cloud data. The following example describes the case of public key {N, X}:

ESS Key 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 $X' = X^{-1} \bmod L = 247^{-1} \bmod 36 = 7$

Compute GCD(Y,M) = GCD(851, 396) = 1

Calculate Inverse $Y' = Y^{-1} \bmod M = 851^{-1} \bmod 396 = 47$

Perform Multiplication Z = X' * Y' = 329

Check GCD (Z,L) = = 1, GCD(329, 36) = 1

So, calculate

$N = Z \bmod L = 329 \bmod 36 = 5$

$d = N^{-1} \bmod L = 5^{-1} \bmod 36 = 29$

ESS Encryption & Decryption

Let us assume plaintext m = 32.

Encryption, ciphertext $C = m^N \bmod X$

$C = 32^5 \bmod 247 = 223$

Decryption, plaintext $m = C^d \bmod X$.

$m = 223^{29} \bmod 247 = 32$.

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

ESS 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 $X' = X^{-1} \bmod L = 77\text{-}1 \bmod 30 = 23$

Compute GCD(Y,M) = GCD(221, 48) = 1

Calculate Inverse $Y' = Y^{-1} \bmod M = 221^{-1} \bmod 48 = 5$

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

$N = Z \bmod M = 115 \bmod 48 = 19$

$d = N^{-1} \bmod M = 19^{-1} \bmod 48 = 43$

ESS Encryption & Decryption

Let us assume plaintext m = 32.

Encryption, ciphertext $C = m^N \bmod Y$,

$C = 32^{19} \bmod 221 = 111$

Decryption, plaintext $m = C^d \bmod Y$,

$m = 111^{43} \bmod 221 = 32$.

EXPERIMENTAL RESULTS

ESS for a sole proprietor Cloud data is generated for simulation purposes using Java BigInteger library methods (Neal R Wagner 2003) and by deploying Eucalyptus on a 2.50 GHz Intel ® CoreTM i5-3120M Processor with 8 GB RAM. Five clusters were randomlyformed, each with four nodes(owners), and each cluster was assigned to a community. To produce a public and private key, each cloud node will call the ESS_Keygen() function. Then, each node in the cluster will exchange its public key with other nodes in the cluster. The proposed ESS system's security and performance have been evaluated using this cloud arrangement. The SS and ESS are secured using a brute force attack and a number field sieve - integer factorization technique. To determine the algorithm's overall execution time, the various performance indicators for ESS are analysed.

PERFORMANCE ANALYSIS - VARIABLE FILE SIZE

The proposed ESS cryptosystem's performance for single owner cloud data is analyzed with cloud data. For effectivecomparison, small input filesizes are given to SS & proposed ESS variable input file sizes. When compared to the Schmidt Samoa cryptosystem, execution time for encryption & decryption versus proposed Enhanced Schmidt Samoa cryptosystem for single owner encryption and decryption ranges from 8KB to 1024 KB, with key sizes remaining constant at 16 bits and 32 bits.

In SS, the key value for 16 bits is taken with p=257, q=349, N=23051101, and d=19189. In ESS, the key value for 16 bit is taken with p=367, q=379, r=389, s=401, N=4099, and d=8699.

Table 1 shows execution time of Schmidt Samoa cryptosystem and Enhanced Schmidt Samoa cryptosystem for single owner cloud data with key size 16 bits.

Table 1: Execution time in SS and ESS with key size 16 bits for single owner cloud data

File size (KB)	SS for variable key size			ESS 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		264	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 key value for 32 bit is taken with p=73529, q=73547, N=506233051, and d=1475417611. For ESS, the key value for 32 bit is taken with p=73529, q=73547, r=73553, s=73561, N=2391448019, and d=1564100563.

Table 2 the execution time of the Schmidt Samoa cryptosystem and Enhanced Schmidt Samoa cryptosystem for single owner cloud data with key size 32 bits.

Table 2: Execution time in SS and ESS with key size 32 bits for single owner cloud data

File size (KB)	SS for variable key size			ESS 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

The inference of Tables 1 and 2 are as follows: The computation cost for the key generation of ESS is high compared to SS, because of the complex operations. In the SS key generation, the computation cost depends on two multiplications, one inverse, and one lcm operation. The computation cost depends on two lcm operations, three multiplications, and two inverse operations in ESS key generation. But related to encryption and decryption time, ESS performs faster than SS. The computation cost of ESS is less because of the small size of keys used for encryption and decryption, compared to SS. Further, the comparative analysis has been performed with Schmidt Samoa, RSA, Paillier, and Enhanced Schmidt Samoa cryptosystems. The experiments have been conducted for a fixed key size of 1024 bits with various file sizes in MegaBytes (MB). Figures 4 and 5 show Schmidt Samoa, RSA, Paillier, and Enhanced Schmidt Samoa cryptosystems performance related to computation costs of encryption and decryption for variable file sizes and fixed key size of 1024 bits. The results show that ESS performs faster compared to Schmidt Samoa, RSA, and Paillier cryptosystems.

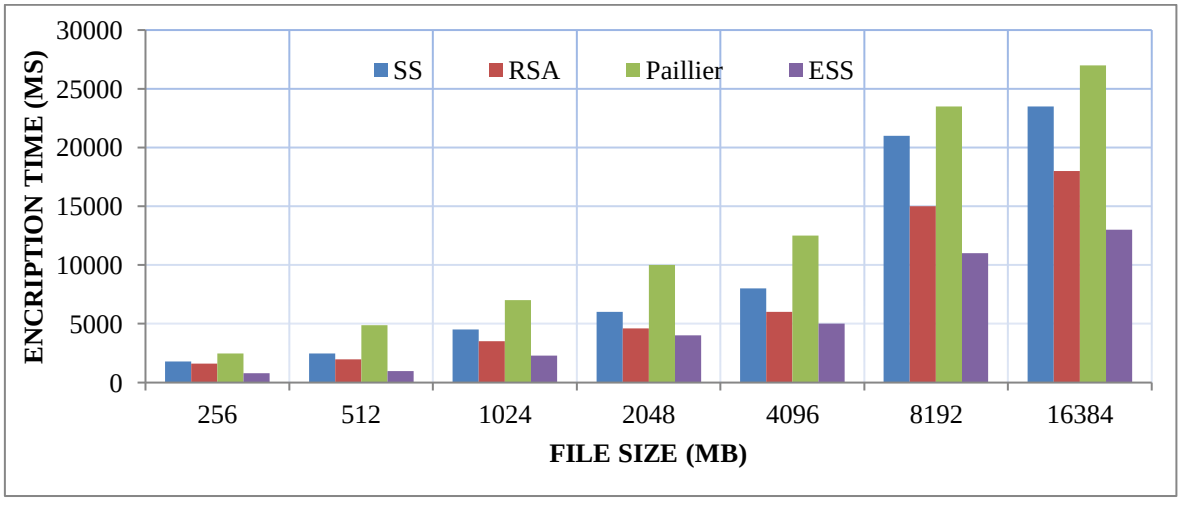

Figure 4: Encryption time for single owner cloud data (key size - 1024 bits)

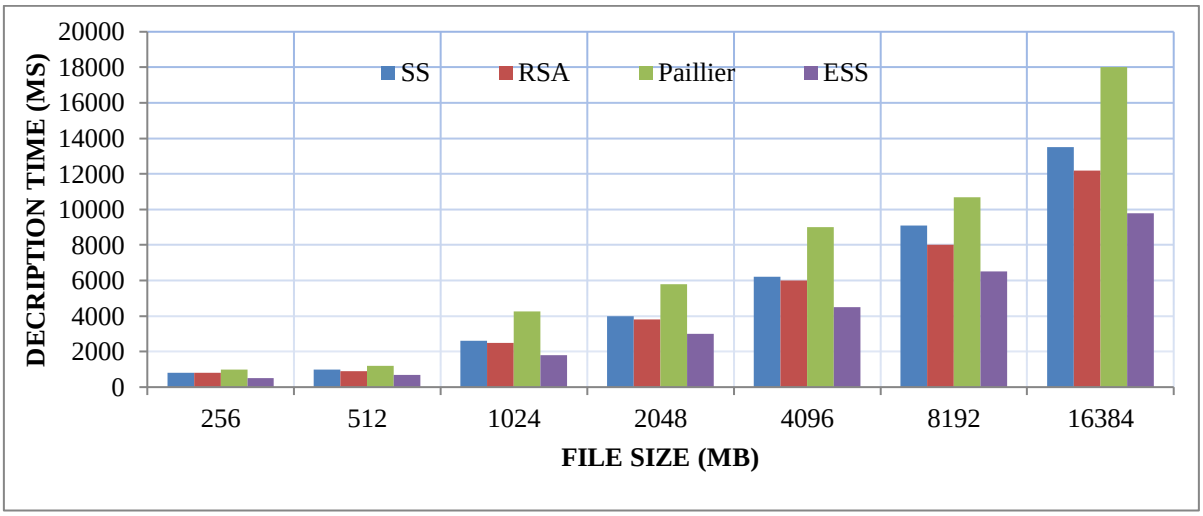

Figure 5: Decryption time for single owner cloud data (key size – 1024 bits)

PERFORMANCE ANALYSIS - VARIABLE KEY SIZE

The performance of the Enhanced Schmidt Samoa cryptosystem has been further analyzed with variable key sizes. Table 3 and Table 4 shows the key generation time of the Schmidt Samoa cryptosystem and enhanced Schmidt Samoa cryptosystem for single owner cloud data. The public key & private key are calculated for different private key sizes of 4, 8, 12, 16, 24, and 32 bits for analysis.

Table 3: Key generation time in SS for single owner cloud data

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	169	32
16	257	349	2301101	19189	47
24	13499	13513	1363151905	10757113	68
32	73529	73547	506233051	1475417611	94

The results reveal that, due to the complicated processes required, the calculation cost of Enhanced Schmidt Samoa key generation increases as the private key size grows when compared to SchmidtSamoa. The ESS Cryptosystem employs four huge ordered prime numbers rather than two random prime numbers, lengthening the attack time. The calculation of the private key 'd' is affected by either X or Y. As a result, calculating 'd' is not straightforward.

As a result of the prime numbers, the key generation task includes a decision-making process, which takes longer than Schmidt-Samoa. Keys will be produced just once for secure communication and

resource access and then used for encryption and decryption as many times as feasible. The longer key generation period, longer the invaders' time to breach the ESS cryptosystem.

Table 4 Key generation time in ESS for single owner cloud data

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	15641005633	124

Table 5 represents the encryption time and decryption time for Schmidt Samoa and Enhanced Schmidt Samoa in microseconds for single owner cloud data with input file size is 64 KB, keys in Table 3 & Table 4, respectively. Similarly, Table 6 provides a comparison with 128 KB. In SS, encryption is performed with plaintext &public key. Thepublic key is computed by multiplying three large prime numbers. The public key is used as exponentiation and modulus to the plaintext. Since a large number has been used as exponentiation, it takes more time to perform encryption.

In SS, decryption is performed with ciphertext and private keys. The value of ‘L’ is computed by taking the least common multiplier between two large numbers. The private key has been computed by taking the modular inverse of public key with L as the modulus. The private key has been used as an exponentiation to the ciphertext. Then, the result of exponentiation is taken modulus with the multiplication value of two large prime numbers. Compared to encryption, SS cryptosystem performs faster decryption. The public key in ESS key creation based on four big prime integers is either N, X or N, Y. The values of N, X, & Y are obtained by utilising a least common multiplier, modular inverse, or multiplication operations, which is a small number when compared to public key of SS method. The public key 'N' was used to exponentiate plaintext. The public key 'X' or 'Y' is then used to perform modulus on the exponentiation result. ESS delivers quicker encryption than SS since the public key values are not as big.

In ESS, public key ‘X’ or ‘Y’, private key ‘d’, and ciphertext are used for decryption. The value of ‘L’ has been computed by taking the least common multiplier between two large numbers. The value of ‘M’ has been computed by taking the least common multiplier between another two large numbers. The private key ‘d’ has been computed by taking the modular inverse of public key ‘N’ over ‘L’ or ‘M’. Private key ‘d’ is used as exponentiation to the ciphertext. Then, the result of exponentiation is taken modulus with public key ‘X’ or ‘Y’. Since the value of private key ‘d’ & public key ‘X’ or ‘Y’ is small compared to SS, ESS performs faster decryption compared to SS.

ESS has a smaller public and private key size than SS. As a result of the key size utilized in encryption and decryption techniques, the computing cost of ESS is lower than that of SS. According to this performance analysis, when the private key size increases, the ESS encryption and&decryption execution time for single owner cloud data will be smaller than the SS encryption and decryption execution time.

Then, the comparative analysis has been performed with fixed file size and variable key sizes among Schmidt Samoa, RSA, Paillier, and Enhanced Schmidt Samoa cryptosystems. Figure 6 and 7 show Schmidt Samoa, RSA, Paillier, and Enhanced Schmidt Samoa cryptosystems performance related to computation costs of encryption and decryption for variable key sizes and fixed file size of 3 GB. The results show that ESS performs faster compared to Schmidt Samoa, RSA, and Paillier cryptosystems.

Table 5: Encryption and decryption time of SS and ESS for single owner cloud data with the file size 64 KB

Key size	**SS for variable keysize and file size 64 KB**		**ESS for variable keysize 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 6: Encryption and decryption time of SS and ESS for single owner cloud data with file size 128 KB

Key size	**SS for variable key size and file size 128 KB**		**ESS 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

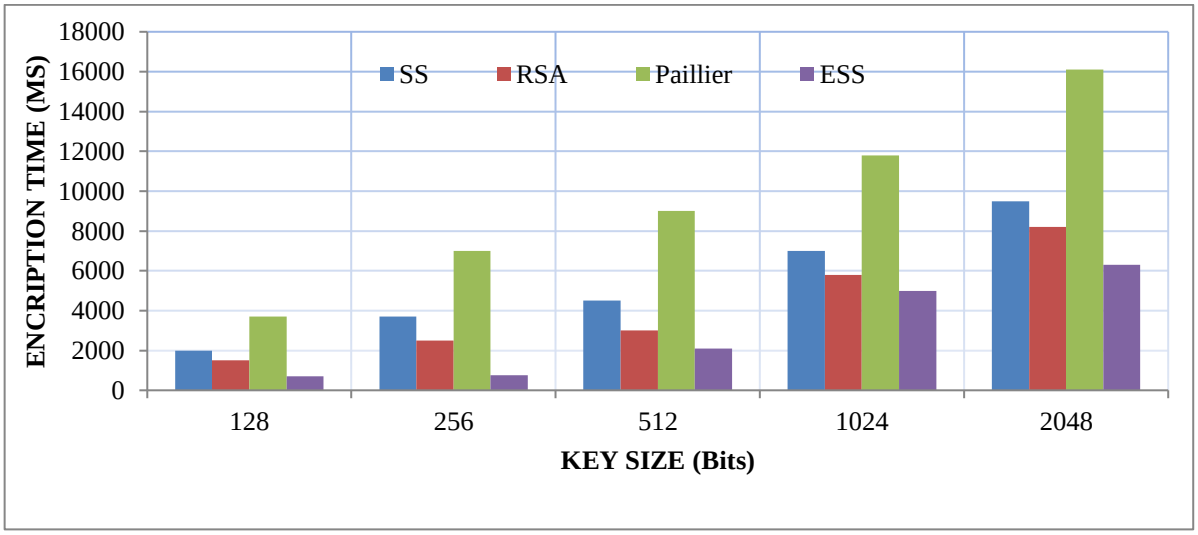

Figure 6: Encryption time for single owner cloud data (file size - 3 GB)

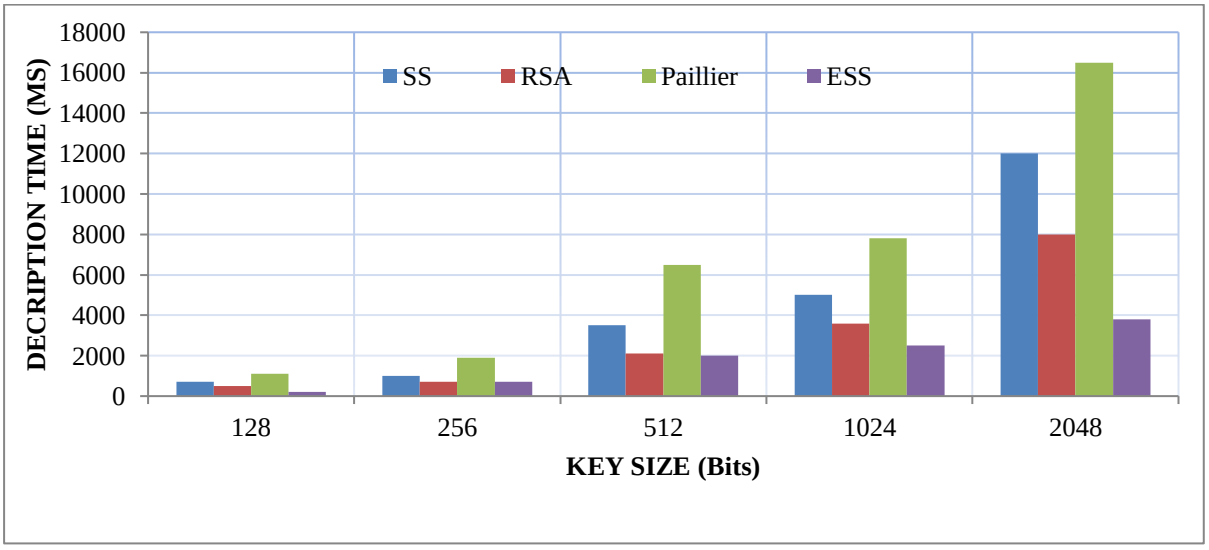

Figure 7 Decryption time for single owner cloud data (file size - 3 GB)

CONCLUSION

In the research work, Improved Secure Cloud Data Storage Framework (ISCDSF) was proposed and implemented for ensuring data confidentiality, integrity, and availability of single owner and multi-user data. The research work focuses on ensuring data confidentiality for single owner data: Enhanced Schmidt Samoa (ESS) cryptosystem is proposed to protect and ensure single owner data confidentiality in the cloud. By raising the complexity of the key generation method, the security of

the cryptosystem is increased. A single owner produces a key only once. However, a single owner regularly executes upload and download activities. The mathematical argument demonstrates that the encryption and decryption procedures are inverse of one another. The studies are conducted out in a cloud environment, and the findings show that the ESS cryptosystem outperforms the SS cryptosystem.

REFERENCES

1. Lu Ming (2007) 'Symmetric-key cryptosystem with DNA technology', Science in China Series F: Information Sciences, Springer, vol. 50, no. 3, pp. 324-333.
2. Hovav Shacham (2008) 'Compact proofs of retrievability', Journal of Cryptology, Springer, vol. 26, no. 3, pp. 442-483.
3. Chris Erway, C, (2009) 'Dynamic provable data possession', Proceedings of the 16th ACM conference on Computer and communications security, ACM, pp. 213-222.
4. Lai X ue Jia, (2010) 'Asymmetric encryption and signature method with DNA technology', Science China Information Sciences, Springer, vol. 53, no. 3, pp. 506-514.
5. Giuseppe Ateniese, (2011) 'Remote data checking using provable data possession', ACM Transactions on Information and System Security, vol. 14, no. 1, pp. 1-34.
6. Peng Y ong, (2012) 'Secure cloud storage based on cryptographic techniques', The Journal of China Universities of Posts and Telecommunications, Elsevier, vol. 19, no. 2, pp. 182-189.
7. Cong Wang, (2013) 'Privacy-preserving public auditing for secure cloud storage', IEEE Transactions on Computers, vol. 62, no. 2, pp. 362-375.
8. Zhang, (2014), 'Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates', IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 9, pp. 2234-2244.
9. Chang Liu (2015), 'MuR-DPA: Top-down leveled multi-replica Merkle hash tree based secure public auditing for dynamic big data storage on cloud', IEEE Transactions on Computers, vol. 64, no. 9, pp. 2609 2622.
10. Samiha Marwan, (2016), 'DNA-based cryptographic methods for data hiding in DNA media', Biosystems, Elsevier, vol. 150, no. 1, pp. 110 118.
11. Su Peng, (2017), ' Identitybased public multi-replica provable data possession', IEEE Access, vol. 5, pp. 26990-27001.
12. Xing-Quan Fu, (2018), 'Image encryption-then-transmission using DNA encryption algorithm and the double chaos', IEEE Photonics Journal, vol. 10, no. 3, pp. 1-15.
13. Su Peng, (2019), 'Efficient, dynamic and identity-based Remote Data Integrity Checking for multiple replicas', Journal of Network and Computer Applications, Elsevier, vol. 134, pp. 72-88.
14. Hossein Nematzadeh, (2020), 'Binary search tree image encryption with DNA', Optik-International Journal for Light and Electron Optics, vol. 202, pp. 1-10.
15. Viswanath and Krishna etal. (2021) "A Hybrid Cryptographic Model Using AES and RSA for Sensitive Data Privacy Preserving" Webology, Volume 18, Special Issue on Current Trends in Management and Information Technology, October, 2021 Received May 06, 2021; Accepted August 08, 2021 ISSN: 1735-188X DOI: 10.14704/WEB/V18SI05/WEB18219.

16. Ramachandra, (2022) "An Efficient and Secure Big Data Storage in Cloud Environment by Using Triple Data Encryption Standard" Big Data and Cognitive Computing 6, no. 4: 101.
17. Cuzzocrea, A.; Karras, P.; Vlachou, A. Effective and efficient skyline query processing over attribute-order-preserving-free encrypted data in cloud-enabled databases. Future Gener. Comput. Syst. 2022, 126, 237–251.
18. A, Reyana, Sandeep Kautish, Sapna Juneja, Khalid Mohiuddin, Faten Khalid Karim, Hela Elmannai, Sara Ghorashi, and Yasir Hamid. 2023. "Enhanced Cloud Storage Encryption Standard for Security in Distributed Environments" Electronics 12, no. 3: 714.
19. K atja Schmidt Samoa 2006, 'A new rabin-type trapdoor permutation equivalent to factoring', Electronic Notes in Theoretical Computer Science, Elsevier, vol. 157, no. 3, pp. 79-94.
20. Neal R Wagner 2003, The Laws of Cryptography with Java Code, The University of Texas, San Antonio.