International Journal of Computer Applications (0975 – 8887)
Volume 136 – No.6, February 2016

Public Encryption Techniques for Cloud Computing: Randomness and Performance Testing

Ayman Helmy Mohamed
Computer science Dept.
faculty of computer and information
Helwan University, Egypt

Aliaa A.A. Youssif
Computer science Dept.
faculty of computer and information
Helwan University, Egypt

Atef Z. Ghalwash
Computer science Dept.
faculty of computer and information
Helwan University, Egypt

ABSTRACT

Cloud computing becomes the next-generation architecture of IT Enterprises. Cloud computing is a technology. It enables clients to use high-end services in a form of software as a service. These reside on different servers all over the world. There are many security threats in cloud computing. Data security is one of them. Data security raises client concerns. There are many issues of data security. It named maintenance of data integrity, data hiding and data safety. These threats dominate clients concerns when the issue of cloud come up. Cloud computing has a big data. Traditional encryption method is time-consuming in this environment. Cloud computing has a single security architecture. It has also many customers with different demands of security. In this case, data security is considered one of the most important issues in cloud computing.

The proposed work focused on accessing data securely in cloud and desktop environment. It depends on public key cryptosystem. Generally, Data security is an important factor for both cloud computing and traditional desktop applications. Client needs to have the highest possible level of privacy. Public key cryptosystem is a good candidate for this purpose. It plays a vital role in cloud computing security and desktop. The paper presents an evaluation for selected public key cryptosystem techniques. It named ElGamal cryptosystem, modified chaotic cryptosystem and chaotic cryptosystem. A modified chaotic cryptosystem was presented. The modification enabled the same message to have different cipher version. The experiments implemented at two independent platforms namely desktop computer and Amazon EC2 Micro Instance cloud computing environment.

In this paper, the selected algorithms compared according to randomness testing. A standardized NIST statistical testing is used in both desktop environment and cloud computing environment. The algorithms are implemented using python charm Cryptography framework. Simulation results are shown to demonstrate the effectiveness of each algorithm. NIST statistical tests are used to determine suitable technique for cloud computing environment and desktop. It also used to study the performance of the selected encryption techniques in both environment.

General Terms

Cloud computing, Security. NIST statistical testing, chaotic, a Modified chaotic, ElGamal

Keywords

Cloud computing, ElGamal cryptosystem, lattice based cryptosystem, chaotic cryptosystem.

1. INTRODUCTION

The security related strongly to randomness. Randomness has high statistical quality. High statistical quality can withstand analysis cryptographic system. The system can compromised due to the inadequate randomness generators. Randomness is the outcome of probabilistic process. It produces independent (lack of correlation), uniformly distributed (lack of bias) and unpredictable values (lack of predictability). The statistical analysis of random sequence is highly important along with the application of statistical tests. They assess the outcome of randomness generator. The tests evaluate the random properties of a sequence. Good random properties must be unpredictable, irreducible and not allow prediction of former or subsequent values [1].

Random number generator classified in three categorization namely true random number generator [2], pseudorandom number generator, and unpredictable random number generator. True random number generator depends on physical phenomena. It extracts randomness by sampling and digitizing physical phenomena. Pseudorandom number generator depends on the initial parameter. It extracts randomness from initial value that named seed. It provides modality to generate random sequences using software method. Unpredictable random number generator mimic a practical approximation of true random number generator. It extracts randomness from easily available devices. It based on the behavior of hardware devices. The intervention in the generation process disturbs the internal state. It is very difficult to predict or model to produce the output.

Unpredictable random produces a stream of zeros and ones that divided into sub streams or blocks of random numbers. The need of randomness arises in many cryptographic applications. For example, keys are generated in a random fashion in a common cryptosystems. Many cryptographic protocols also require randomness inputs at various points. It used as auxiliary quantities in generating digital signatures, or for generating challenges in authentication protocols [3]

The National Institute of Standards and Technology (NIST) developed a statistical test suite. It named a statistical package. It consists of 15 tests [4] that were developed to test the randomness of arbitrary long binary sequences produced by either hardware or software. The sequence depends on random or pseudorandom number generators. These tests focused on different types of non-randomness properties that could exist in a sequence. It helps in detecting the deviation of a binary sequence from randomness. This deviation may occur due to a poorly designed generator or to anomalies that appear in the binary sequence [5].

The paper was evaluated some public key cryptography. It named ElGamal cryptosystem, a modified chaotic cryptosystem and lattice based cryptosystem. This evaluation has been performed for those encryption algorithms according to randomness testing. The NIST statistical testing used in both cloud computing and traditional desktop environments. Performance analysis was conducted. The performance analysis was on encryption/decryption speed and rejection rate per test for those encryption algorithms. The tests are implemented in both cloud computing and traditional desktop environments. This evaluation is used to determine the most recommended technique in both environment. It also introduce analysis performance of selected public encryption techniques.

The rest of this paper is organized as the following. Literature review is presented in section 2. NIST statistical tests for randomness are presented in section 3.security algorithms is presented in section 4. Finally, results, conclusion and future scope are presented in section 5, and 6 respectively.

2. LITERATURE REVIEW

The most basic property of evaluating block and stream ciphers is to pass statistical randomness testing. The author of [3] proposed a framework. It tests statistically any cryptographic algorithm. It used to evaluate the randomness of cryptographic algorithms. It depends on a .dll file. It accesses the encryption function, the decryption function and the key schedule function of the cipher. It applied nine tests to evaluate AES candidate block ciphers. The author evaluated Tiny Encryption Algorithm (block cipher), Camellia (block cipher) and LEX (stream cipher). The tests focused on finding any detectable correlation between plaintext-cipher text pairs, any detectable bias due to single bit changes to either a plaintext or a key, and many others. It shows that LEX, Camellia and TEA algorithms have good statistical results for all the tests applied. The framework does not evaluate memory usage, CPU time, performance and security.

In paper [6], it mentioned that cloud computing has a single security architecture but has many customers with different demands. It focused on data storage security in the cloud and the desktop. The paper evaluates eight modern encryption techniques. These named RC4, RC6, MARS, AES, DES, 3DES, Two-Fish, and Blowfish. It was evaluated at two independent platforms. It named desktop computer and Amazon EC2 Micro Instance cloud computing environment. The evaluation used to perform randomness testing. It used NIST statistical testing in cloud computing environment and desktop. The Simulation results shown no strong indications of statistical weaknesses for eight encryption algorithms in both environments, but some differences between algorithms appeared. It shows that Random Excursions Variant test and Random Excursions test are not applicable for the tested algorithms. In Amazon EC2, Blowfish RC6, AES and DES results were outperform other-encryption methods and AES. It does not tested the public key cryptosystem. It also does not consider the key generation in the process of evaluation.

In paper [7], a symmetric key encryption and encoding are analyzed. It also analyzed various algorithms and compared them based on security and performance. It was recommended a solution to protect and to verify users' sensitive data using cryptographic techniques. It has analyzed AES, DES & 3DES symmetric encryption techniques on one side. On the other side, it has analyzed MD5 and SHA-256 encoding techniques. It shows AES is a good candidate for symmetric key encryption. It outperforms DES and 3DES since they produced overhead. It shows also MD5 faster for encoding. It does not tested asymmetric key encryption technique along with social networking as a datasets. In paper [8], a simple protection model was proposed for cloud computing. It used AES algorithm to encrypt user's data before it launched in the cloud. It helps users to choose infrastructure according to their security requirements.

Cloud provides high data storage but there is always a problem of security of data stored in cloud. In this paper [9] a proposal was discussed to solve cloud storage security problem. It verifies the integrity of outsourced data. It supports detection of anomaly and dynamic data operations. It has a drawback in metadata size .it applied also low order hybrid chaotic sequence tag generation method

3. NIST IN CLOUD COMPUTING

Clouds are massively a complex system. This complexity of cloud computing creates many issues related to security as well as all aspects of Cloud computing. Data security is one of the most import issue in cloud computing. Cloud computing has a single security architecture on the other hand they have many customers with different security requirements. Statistical hypothesis testing is used to evaluate whether an experimental set of data fits with a given hypothesis. It is largely adopted in testing random number generators. The tested data is a long stream of bits "the discerning between a random stream and the encrypted signal is computationally hard problem". The NIST test can be applied for all randomness generators [4].

The NIST Test Suite is a statistical package consisting of 15 tests. It is developed to test the randomness of binary sequences. It is produced by either hardware or software. These tests focus on a variety of different types of non-randomness. These can exist in a sequence. Some tests are decomposable into a variety of subtests. The statistical test is based on hypothesis testing. The hypnosis has (H_0) and (H_a). (H_0) named the null hypothesis. (H_a) named the alternative hypothesis. The decision is built on the value of p-value. If p-value <= H_0 then the decision will be as follow, the test result has strong evidence to reject H_0. If p-value> H_0 then the test do not have a strong evidence to reject H_0 [4].

There are two types of error with statistical tests. Type I error, it means H_0 is true and the decision will be as follow you do not have a strong evidence to reject H_0. Type II error, it means H_0 false and the decision will be as follow, the test has a strong evidence to reject H_0. The errors means that type I is false positive and type II is true negative.

4. SECURITY ALGORITHMS

Encryption algorithm plays a vital role in securing data communications. It is the fundamental tool for protecting data. Encryption algorithm converts the data into unreadable form. It used keys to perform the process. The user have the key able to decrypt the data. In Symmetric key encryption, only one key is used to encrypt and decrypt the data. Another technique is named symmetric key encryption. It used two keys- private and public keys. Public key is used for encryption. Decryption used private key. Figure 1 shows some of the symmetric and asymmetric algorithms.

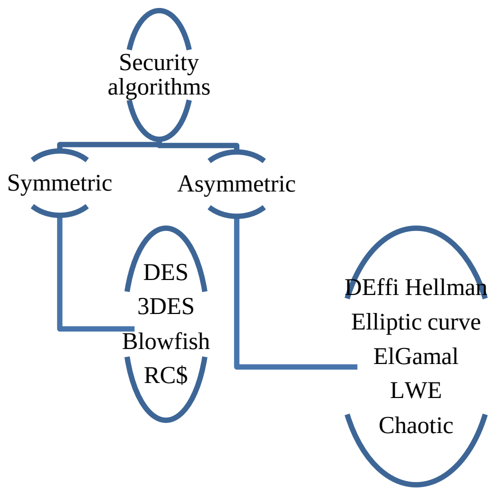

Figure 1 Sample of Security algorithms

4.1 Elgamal Cryptosystem

ElGamal [10] [11] is a public key encryption algorithm. It based on the discrete logarithmic problem for F_p^*. The security hardness of ElGamal depends on solving the following discrete logarithms of $b^l \equiv c \, mod \, p$, in a large prime modulus. Alice begins by publishing information consisting of a public key and an algorithm. The public key is simply a number, and the algorithm is a method by which Bob encrypts his messages using Alice's public key. Alice hides her private key, which is another number. The private key allows Alice, and only Alice, to decipher messages. These messages have been encrypted using her public key.
The algorithm works as follow:

Step 1: Public Parameter Creation; a trusted party chooses and publishes a large prime p and an element $g \; modulo \; p$ of large (prime) order.

Step 2: Key Creation; Chooses private key named "a" where $1 \le a \le p-1$. Computes
$A = g^a (mod \, p)$. (1)
Publishes the public key (p, g, A).

Step 3: Encryption; Chooses plaintext m. Chooses random key k. Uses Alice's public key A to compute $c_1 = g^k \, (mod \, p)$ and $c_2 = mA^k (mod \, p)$. Sends cipher text (c1, c2) to Alice.

Step 4: Decryption; Compute $(c_1^a)^{-1} \cdot c_2 \, (mod \, p)$. This quantity is equal to m.

In ElGamal cryptosystem, the plaintext is an integer m between 2 and $p-1$, while the cipher text consists of two integers c_1 and c_2 in the same range. Thus, in general it takes twice as many bits to write down the cipher text as it does to write down the plaintext. Therefore ElGamal has a 2 to 1 message expansion [10] It works as follow Alice chooses $p = 113, g = 2, a = 71$. She Computes $A = 2^{71} \equiv 111$ (mod 113). Her public key is (p, g, and A) = (113, 2, 111), and her private key is a= 71. Bob wants to send the message "Z" (90 in ASCII) to Alice. He chooses a random integer k = 23 and encrypts M = 90 as $(r,t) = (g^k \; ,A^k.M) \equiv (2^{23}, 111^{23}.90) \equiv (53, 89) \; (mod \; 113)$. He sends the encrypted message (53, 89) to Alice. She receives the message (r, t) = (53, 89), and using her private key a = 71 she decrypts to $tr^{-a} = 89 \cdot 53^{-71} \equiv 89 \cdot 53^{112-71} \equiv 89{\cdot}81 \equiv 90$ (mod 107).

4.2 Chaotic Cryptosystem

The chaotic cryptosystem depends on defining the mapping scheme for trajectory, choosing valid initial condition and parameters. The chaotic cryptosystem generates random number sequence. The generated sequence random number used to encrypt the message. There are many chaotic systems namely: Lorenz map, Rossler map, logistic map, Tent map, and Chebyshev map.

The Lorenz map is simple differential equations. The solution of these equations recursively can demonstrate many of the principals of chaotic systems.it models the weather. The Lorenz system consists of the following equations:-

$$\frac{dx}{dt} = \sigma(y - x) \quad (2)$$
$$\frac{dy}{dt} = rx - y - xz \quad (3)$$
$$\frac{dz}{dt} = xy - bz \quad (4)$$

Variables: [x, y, z] (initial conditions) must be specified by the user.
Parameters: $\sigma, r \; and \; b.$
The parameters can change the shape of the resulting trajectories. The Lorenz well known chaotic attractor named Lorenz Butterfly. The lower value r (say 22) will graph a fixed point for the same a and b settings. The higher value r (say r 330) will begin to see the Lorenz limit cycle.

The Rossler map is a simple differential equations. It likes the Lorenz system. The Rossler equations are:

$$\frac{dx}{dt} = -y - z \quad (4)$$
$$\frac{dy}{dt} = x + ay \quad (5)$$
$$\frac{dz}{dt} = a + z(x - c) \quad (6)$$

Variables: [x, y, z]
Parameters: a, and c

The Logistic map displays very specific chaotic effects. The system equation of chaotic is simple. The logistic equation is:
$f(x) = rx(1 - x) \quad 0 \le r \le 4$ (7)
Variable: x
Parameter: r

The tent map is very similar to the logistic map. It displays some very specific chaotic effects. The tent map equation is:

$$f(x) = \begin{cases} 2tx & x < \frac{1}{2} \\ 2t(1-x) & x \ge \frac{1}{2} \end{cases} \quad 0 \le t \le 1$$
(8)
Variable: x
Parameter: t
Chebyshev [3] [12]chaotic maps is a source of chaotic dynamic for a long time. Its commutative properties enabled it to develop a public key cryptosystem. The commutative properties can represented as follow $T_r(T_s(x)) = T_s(T_r(x)) = T_{rs}(x)$. A modified public key encryption based on chebyshev and logistic maps was proposed as follow:-
Chebyshev equation is
$T_n(x) = 2.x.T_{n-1}(x) - T_{n-2}(x) \quad for \; n > 2, T_0(x) = 1 \; and \; T_1(x) = x$ (9)

Logistic map equation is $x_{n+1} = 3.99 * x_n(1 - x_n)$ (10)

"A", in order to generate the keys, do the following:

1. Generate a random fraction as an initiate seed (x)
2. Generate random a large integer numbers (S)
3. Generates an integer seed using logistic map equation (10) which is based on (S, x) (seed).
4. computes $T_s(seed)$ based on seed and S
5. A sets her public key to (seed, $T_s(seed)$) and her private key to (S).

B, in order to encrypt a message, does the following:

1. Obtains A's authentic public key (seed, $T_s(seed)$).
2. Represent M as an integer number
3. Generates a large integer r.
4. Computes $T_r(seed)$
5. and Enc. =M $\oplus T_r(T_s(seed))$
6. Sends the cipher text (Enc, $T_r(seed)$). To A

A, to recover the plaintext M from the cipher text C, does the following:

1. Uses her private key (s) to compute $T_s(T_r(seed))$

Recovers M by computing $M=Enc \oplus T_s(T_r(seed))$.

5. EXPERIMENTS AND RESULTS

testing environment was desktop and cloud computing respectively. The desktop was laptop. It has Intel core 2.26 GHZ with 512 KB cache and 2GB RAM. Microsoft windows 7 was used. The cloud was Amazon EC2 medium instance. It has Intel AVX 2.5 GHZ, 4 GB RAM. Microsoft operating system was used. It has two virtual core CPU. The simulation has developed using python language. It was Python 2.7.6. It used NumPy plugin for the mathematical calculation. Charm cryptography framework was used in implementation. It used to implement tested cryptography algorithms. All graphs in this paper were also made in Microsoft Excel 2013.

It frequently used message spaces for public-key encryption. The messages were 180 randomly generated messages. The message were 32, 64, 128 bit. It was generated using Pythons build in random number generator. Thus, the messages were set from the beginning and throughout all of the experiments. The testing was done 180 times per cryptosystem. The keys were pre-generated for each algorithm. The tested cryptosystem namely ElGamal, modified chaotic and logistic chaotic cryptosystem. The encrypted messages passed through 15 NIST statistical tests. These tests divided into two categories parametric and non-parametric.

It was noticed that the non-parametric tests are almost normal for all mentioned above algorithms. It was not notice strong statistical weaknesses for tested algorithms. The percentage of failure in the test almost less than 1%. In all non-parametric tests. It was noticed that the higher p-value is the best. It was noticed that the longest messages generates higher p-value. The longest message have randomness features than the shortest message.

The NIST statistical tests used to test the encryption results of each algorithm. Table 1 depicts the average recorded results per each cryptosystem per each non-parametric NIST statistical test. The recorded results is the mean value for each algorithm per test. The result recorded for messages in different size. It was 32, 64 and 128 bit length.

Table 1 averaged non-Parametric tests results for ElGamal, modified chaotic and chaotic for 128-bit messages

Tests	Modified chaotic	ElGamal	Chaotic
Binary matrix rank	0.62339	0.59731	0.60299
Lempelziv compression	0.37847	0.30847	0.27847
Longest run ones	0.35588	0.25588	0.49289
Monobit frequency	0.61835	0.62675	0.65881
Random excursions	0.69106	0.57838	0.57661
Random excursions variant	0.74000	0.64946	0.68164
Runs	0.60408	0.50785	0.58502
Spectral	0.63070	0.50824	0.45578

It noticed that all algorithms passed non-parametric tests. It noticed that a modified chaotic cryptosystem almost outperforms chaotic and ElGamal cryptosystem. It also noticed that chaotic cryptosystem outperforms ElGamal cryptosystem.

Table 2 averaged non -parametric tests results for ElGamal, modified chaotic and chaotic for 64-bit messages

Tests	Modified chaotic	ElGamal	Chaotic
Binary matrix rank	0.5108	0.476	0.34567
Lempelziv compression	0.31246	0.23547	0.24567
Longest run ones	0.39512	0.23457	0.34567
Monobit frequency	0.30000	0.21015	0.24567
Random excursions	0.58357	0.55235	0.34567
Random excursions variant	0.53400	0.46847	0.39456
Runs	0.41116	0.47757	0.34567
Spectral	0.48749	0.46634	0.34567

Table 3 averaged non -parametric tests results for ElGamal, modified chaotic and chaotic for 32-bit messages

Tests	Modified chaotic	ElGamal	Chaotic
Binary matrix rank	0.40152	0.37047	0.30451
Lempelziv compression	0.23748	0.04515	0.01522
Longest run ones	0.26504	0.04485	0.12736
Monobit frequency	0.23333	0.26212	0.17966
Random excursions	0.52238	0.33435	0.22074
Random excursions variant	0.50600	0.30068	0.24691
Runs	0.27507	0.28514	0.21648
Spectral	0.37124	0.26019	0.21871

Table 1, 2, and 3 show a comparison between a modified chaotic, chaotic and ElGamal public key cryptosystem. It shows the average P-value per non-parametric tests. The recorded results show the effect of message length on message randomness. The results show uniformity of applied algorithms. It clears that the uniformity of ElGamal and chaotic are nearly the same. It shows the modified chaotic cryptosystem almost outperforms ElGamal and chaotic in non-parametric tests. It was noticed also that chaotic cryptosystem outperforms ElGamal in non-parametric tests.

Table 4 Averaged parametric tests results for modified chaotic, chaotic and ElGamal for 128-bit messages.

Tests	Modified chaotic	ElGamal	Chaotic
Approximate entropy	0.67118	0.64018	0.54882
block frequency	0.65628	0.55736	0.04390
Linear complexity	0.55675	0.57020	0.49043
Maurer's universal statistic	0	0	0
Non overlapping template matching	0.69941	0.57369	0.53277
Overlapping template matching	0.64114	0.47422	0.53123
serial	0.38781	0.37060	0.38714

Table 5 Averaged parametric tests results for modified chaotic, chaotic and ElGamal for 64-bit messages.

Tests	Modified chaotic	ElGamal	Chaotic
Approximate entropy	0.5	0.46321	0.47415
block frequency	0.47915	0.48531	0.41234
Linear complexity	0.54429	0.51460	0.23440
Maurer's universal statistic	0	0	0
Non overlapping template matching	0.64848	0.59344	0.52197
Overlapping template matching	0.54311	0.43359	0.52031
serial	0.3781	0.32060	0.30871

Table 6 Averaged parametric tests results for modified chaotic, chaotic and ElGamal for 32-bit messages.

Tests	Modified chaotic	ElGamal	Chaotic
Approximate entropy	0.36667	0.15440	0.31646
block frequency	0.02638	0.46961	0.13744
Linear complexity	0.0481	0.34468	0.15651
Maurer's universal statistic	0	0	0
Non overlapping template matching	0.40496	0.52085	0.34770
Overlapping template matching	0.31157	0.30661	0.34701
serial	0.15937	0.10686	0.10290

In table 4, 5, and 6 shows the averaged p-value of parametric tests. The parametric tests used to test modified chaotic, chaotic and ElGamal randomness. It was clear uniformity of the recorded results. It was noticed that all algorithms failed to pass Maurer's "Universal Statistical" Test. It was clear that modified chaotic cryptosystem outperforms Chaotic and ElGamal cryptosystem. It was noticed that ElGamal outperforms Chaotic in parametric tests. it shows the message length effect on the randomness of the encryption process. It was clear that all tests need long binary sequence. It required 100-bit length or more in each tests expect Maurer's universal statistic, it requires 1000 bit or more.

The recorded results in parametric and non-parametric tests show almost uniformity of applied tests. It shows good results of modified chaotic cryptosystem over others. It means the encrypted messages using modified chaotic is very close to true random.

Figures 2, 3 show the rejection rates on both environment. The rejection rates represent the number of cases failed to pass. It means p-value of the test less than σ *where* σ *is* .01 in our tests. It was noticed that the rejection rates on cloud computing is less than the rejection rates on desktop environment. It was noticed that the rejection occurred in tests 2, 5, 11 and 12 on desktop. In Cloud computing the rejection occurred in tests 11, and 12. It was noticed the rejection rates exceed the normal rejection rates in tests 11 and 12 in desktop and cloud environment. These tests required large clock cycle to pass. It was also noticed that ElGamal has the highest rejection rate. The rejection ratio in cloud computing is slightly less than the rejection ratio on desktop.

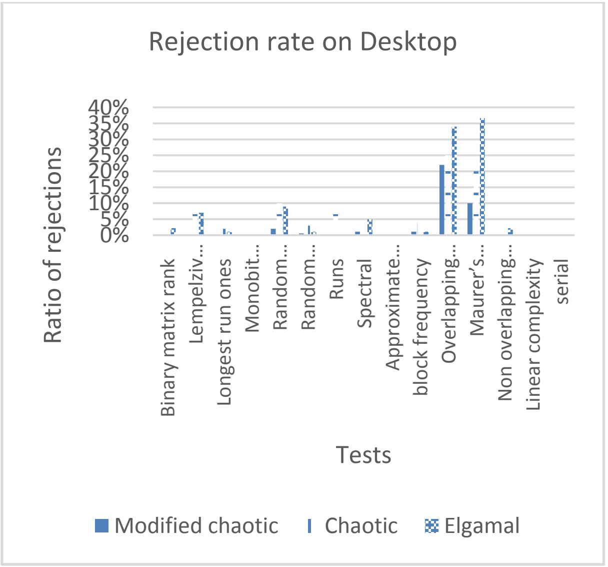

Figure 2 Rejection rates per tests on desktop

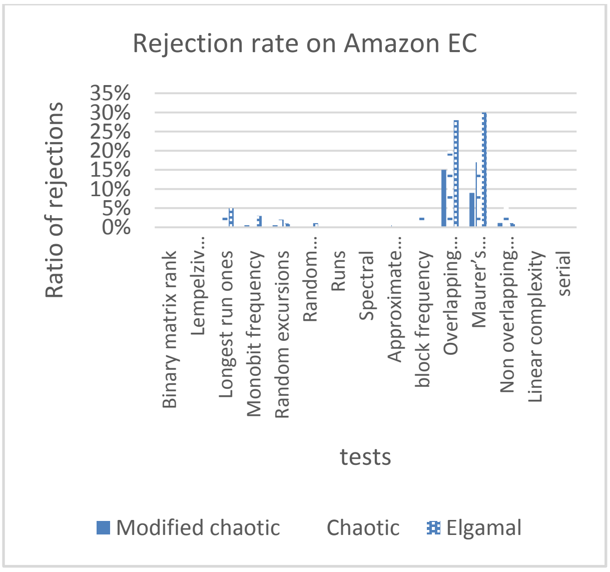

Figure 3 Rejection rates per tests on Amazon EC

In table 7, the chaotic cryptosystem outperforms modified chaotic and ElGamal in the performance in both environment. A chaotic cryptosystem recorded the lowest average encryption time. On the other hand, ElGamal recorded the highest average encryption time. It was noticed that the modified chaotic cryptographic outperforms exponential cryptographic algorithm. It was noticed that the average performance time in cloud computing is less than the average performance measure in desktop environment. Since the amount of data in cloud is very huge. Therefore, it can be encrypted using a modified chaotic or Chaotic. The modified chaotic has the advantage of generating similar messages in different cipher texts.

Table 7 Algorithms performance on desktop and cloud.

Algorithms	Avg. Encryption in Micro Sec.		Avg. Decryption Micro Sec		Avg. Rejection rate	
	Desktop	cloud	Desktop	cloud	Desktop	cloud
ElGamal	6494	4329	1217	900	11%	7%
Chaotic	120	80	120	80	8%	5%
Modified chaotic	220	160	200	150	3%	3%

It shows also the performance of a modified chaotic, ElGamal and chaotic on desktop and cloud-computing environment based on average encryption, decryption time in microsecond along with the average rejection rate. It shows cloud computing recorded a good performance in comparison with desktop. It was also noticed that the average decryption time is less than the average encryption on both environment. The rejection rate is slightly good in cloud computing than desktop environment.

6. CONCLUSION AND FUTURE SCOPE

Cloud computing is an enabler to develop business models. They have a great potential to change IT industry. More clients and organizations will go to cloud for daily computer use as it abstracts the complexity of computing. Security is main concern, which prevents large organization for using cloud. A service provider needs to ensure that applications are safe from all possible attacks. The client, on the other hand needs to ensure that data is safe from intruders. Number of client will increase which would lead to increase in number of keys held at given point of time. Data security can be very good assured by use of linear cryptographic algorithms put the massive amount of data in cloud presents a hindrance to the idea. Developing statistical tests help an organization make an informed decision as to whether cloud computing is currently suitable to meet their business goals with an acceptable level of risks.

The paper reviewed various public encryption mechanism that applied for data security in cloud computing. Three public key cryptosystem was tested. The proposed cryptosystem has the advantage of exponentially, and chaotic difficulty problems respectively. This cryptography named ElGamal cryptography, modified chaotic cryptography and chaotic cryptography. These algorithms were tested on two-different environment namely cloud computing environment and desktop environment. All algorithms almost passed NIST statistical test. The tested algorithms almost achieved uniformity across all tests. The modified chaotic algorithm outperforms ElGamal and chaotic in performance. It also achieved less rejection rate compared to the others. It was noticed that chaotic achieved a good performance compared to exponential cryptography. The tested algorithms can be sorted based on their performance (encryption and decryption) as follow: Chaotic, a modified chaotic cryptosystem, and finally ElGamal cryptosystem. It also can be sorted based on less rejection rates as follow: a modified chaotic cryptosystem, chaotic and finally ElGamal cryptosystem.

From simulation results, it does not have a strong evidence of statistical weaknesses for tested encryption algorithms in both environments. A modified Chaotic Cryptosystem is good option for clients connect to cloud based application with huge data. The cloud needs to store parameters and the system equation to generate the keys. It has the advantage. The same plaintext gives different cipher text each time it is encrypted. It depends on XOR function to encrypt and decrypt data. In Amazon EC2, the evaluation of tested encryption techniques show that a modified chaotic and chaotic results were slightly better than ElGamal encryption methods. All tested algorithms failed to pass Maurer's universal statistical tests. The main reason behind the failure was that the Maurer's Universal Statistical Test requires extremely long sequence lengths.

The message length effect the quality of messages randomness along with random generation of messages. The random generated messages along with more bits is good than random generated message with less bits length. The message sequence length effects the results of NIST tests. A long sequence has a good opportunity to pass all tests if it's random.

Cryptography based on a modified chaotic provided a robust and secured model for development of secured cloud applications. In future, it can use chaotic cryptosystem instead of symmetric and asymmetric cryptosystem implemented in cloud computing. The chaotic maps can used in key generation and encryption process to replace pseudo random generation function. The chaotic can be used to generate random that should be included to generate keys in digital signature along with finding methods for qualitative and quantitative risk analysis.

7. REFERENCES

[1] A.S. C. S. ,. O. C. Kinga MÁRTON, "Generation and testing of random numbers for cryptographic applications," proceedings of the romaanian academy, Series A,, vol. 13, no. 4, p. 368–377, 2012.

[2] P. Z. Li Dejuna, "Research of True Random Number Generator Based on PLL at FPGA," Elsevier Ltd., vol. 29, p. 2432 – 2437, 2012.

[3] B.-C. M. F.-A. V. L. G. Cristina-Loredana Duta, "Randomness evaluation framework of cryptographic algorithms," International Journal on Cryptography and Information Security (IJCIS), vol. 4, no. 1, March 2014.

[4] J. S. L. V. AndrewRukhin, "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," NIST Special Publication 800-22, 2010.

[5] K. U. ,. a. A. H. Song-Ju KIM, "On the NIST Statistical Test Suite for Randomness," information and communication engineers, 2003.

[6] E. M. M. H. S. A.-k. Sherif El-etriby, "Modern Encryption Techniques for Cloud Computing Randomness and Performance Testing," ICCIT, pp. 800-805, 2012.

[7] P. K. H. G. H. P. Krunal Suthar, "Analytical Comparison of Symmetric Encryption and Encoding Techniques for Cloud Environment," International Journal of Computer Applications, vol. 60, no. 19, p. 0975 – 8887, December 2012.

[8] M. B. Abha Sachdev, "Enhancing Cloud Computing Security using AES Algorithm," International Journal of Computer Applications (0975 – 8887), vol. 67, no. 9, April 2013.

[9] D. p. V.V.Jog, "Security of outsourced data in cloud using dynamic auditing," international jornal of computer science Engineering and Techbology (IJCSET), vol. 4, no. 2, pp. 392-394, Dec 2014.

[10] J. P. J. H. S. Jeffrey Hoffstein, "An Introduction to Mathematical Cryptography," Springer: Book, 2008.

[11] P. C. v. O. S. A. V. Alfred J. Menezes, "Handbook of applied cryptography," http://cacr.uwaterloo.ca/hac/.

[12] K. R. a. R. G. K. Prasadh, "Public key cryptosystems based on chaotic Chebyshev polynomials," Journal of Engineering and Technology Research, vol. 1, no. 7, pp. 122-128, October, 2009.