Computational Speeds Analysis of RSA and ElGamal Algorithms on Text Data

A.E .Okeyinka

Abstract- The Elgamal and RSA algorithms are two of the cryptographic techniques that are actively in use for securing data confidentiality and authentication. The energy usage analysis of the two algorithms has been investigated and it was established that RSA is more energy efficient than Elgamal. The goal of this study is to carry out computational speeds analysis of the two algorithms. The methodology employed involves implementation of the algorithms using same programming language, programming style and skill, and programming environment. The implementation is tested with text data of varying sizes. The results obtained reveal that holistically RSA is superior to Elgamal in terms of computational speeds; however, the study concludes that a hybrid algorithm of both the RSA and Elgamal algorithms would most likely outperform either the RSA or Elgamal. It is therefore recommended that efforts at designing a new algorithm from the study of these two algorithms should be considered.

Index Terms- algorithm, cryptography, ElGamal, RSA, speeds

I. INTRODUCTION

Cryptography is concerned with the study of how to keep secrets secret. Its classical task is to provide confidentiality. However, in recent times, the scope of cryptography has expanded beyond issues of confidentiality. Its domain now covers the study of techniques for message integrity, identity, authentication, digital signatures and so forth. The rapid growth of electronic communication means that issues in information security are of increasing practical importance [3]. Many cryptographic algorithms have been developed among which are the following:

A. RSA: This is a public key. It is a bijective function and computationally efficient. It was designed by Rivest, Shamir, and Adleman.

B. ElGamal: ElGamal is a discrete logarithm algorithm. It is a one-way function, and contains no trap door.

C. DES: This is Data Encryption Standard. It uses a 56-bit key and operates on 64-bit blocks of data.

D. HASH: This is also known as 'fingerprint' or 'message digest'. It is used for computing condensed representation of a fixed length message.

E. MD5: This is a 128-bit message digest function, developed by Ron Rivest.

Manuscripts received June 11, 2015; revised July 28, 2015. A.E.Okeyinka,(phone:+2348035776226; email:ae.okeyinka@gmail.com) is with Landmark University, Omu-Aran, Nigeria. He was formerly with Ladoke Akintola University of Technology, Ogbomoso,Nigeria

A cryptographic algorithm is a set of mathematical rules used in encryption and decryption. In addition to securing data being communicated, there is also the need to ensure that the data which is communicated is authentic. A digital signature is a means of ensuring that an electronic document is authentic. "Authentic" in this context implies that the receiver knows the person who created the message, and he knows that the message has not been altered since it was created. A digital signature mechanism consists of an algorithm for generating the signature as well as an associated verification algorithm. Digital signatures are designed to provide authentication and also non-repudiation. In this study, the RSA and Elgamal algorithms including their digital signatures are implemented and compared.

II. RESEARCH MOTIVATION

In addition to creating new algorithms to solve problems that are so far regarded as unsolvable or impractical, the research gradient in computational complexity is also skewing towards algorithm efficiency. It is not enough to invent an algorithm; indeed, considering the computational efficiency of such an algorithm vis-à-vis the existing ones professing to do same task is of great importance. Many cryptographic algorithms abound but they are not equally efficient. In that case there is need to measure and compare their level of computational efficiency. Doing so, would enable us to know which of the algorithms should be used in specific situations for overall maximum efficiency. Furthermore, a reflection on their performance may suggest the need for more study of the algorithms to establish whether or not a more efficient algorithm could be obtained by hybridization or concatenation. So far it has been established that RSA is more energy efficient than Elgamal [6]; however, other performance parameters need to be investigated and studied.

In this paper, RSA and ElGamal algorithms including digital signatures are studied. The choice of these two algorithms is not arbitrary. RSA is a classical technique and most security systems in use today were based on RSA. In short RSA appears to be the most acceptable technique for securing electronic data communication. RSA, was proposed in 1977, [1]. It was patented in the United States by Massachusetts Institute of Technology (MIT) September 20, 1983, [1]. Although the patent for RSA expired September 21, 2000, [4], RSA has become the most popularly implemented public-key cryptosystem [1]. Elgamal on the other hand was proposed in1985, [5]. It is an extension of the Diffie-Hellman key agreement protocol. It is a non-deterministic algorithm, [5]. So the goal of this study is to determine and compare the complexity of RSA and Elgamal algorithms; given that RSA is deterministic and Elgamal is non-deterministic.

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

III. RESEARCH METHODOLOGY

Both the RSA and Elgamal cryptographic algorithms with digital signature are implemented using C# programming language on the same programming environment. Each algorithm consists of three phases:Key generation, Encryption and decryption**,** Signing and verification.

The C# program takes as inputs ten different text data one by one; each character of the text document is converted into its ASCII form and used appropriately in the algorithms in computing cipher text information, which is sent to the receiver by the sender. The cipher text information received by the receiver is decrypted by the module meant for that in order to extract the original message. The length of the text used as input is automatically determined by the C# code. The signature generation, and signature verification module of the code determines the validity of the signature. The execution times of each input text as a whole are observed using the computer internal clock for both Elgamal and RSA algorithms. The execution times are compared to determine which of the two algorithms is more computationally efficient.

IV. RSA AND ELGAMAL ALGORITHMS

The two algorithms (Elgamal and RSA) are presented below.

A. Elgamal Algorithm

Elgamal cryptosystem requires a modular exponentiation operation. The security strength of the cipher is a function of the sizes of the modulus; it is based on the discrete logarithm.

i) Key Generation

This process generates required keys (private key and public key) for both encryption and decryption. The algorithm is stated as follows:

Generate a large Prime number p
Choose a Generator number a subject to the following conditions
$1 < a < p-1$
To ensure that the value of a picked is a generator number, additional conditions have to be considered as follows
Find $Ø = p - 1$
Find all the factors of Ø i.e. (f1, f2…fn).
a is a generator number if and only if $wi = a^{Ø/fi} \bmod p <> 1$, for all qi
Choose an integer x such that
$1 < x < p - 2$
x is the private key
Compute $d = a^x \bmod p$
Public key information = (p, a, d)
Private key = x

ii) Encryption and Decryption Algorithm

The encryption is done using the public key information while the Decryption is done using the private key information.

1) Encryption

The sender receives the public key information only, which will enable her to encrypt
The sender encodes the message m by converting its string representation to its corresponding numerical value.
The sender chooses an integer k such that $1 < k < p - 2$
The sender computes $y = a^k \bmod p$
The sender also computes $z = (d^k * m) \bmod p$
The sender then sends the cipher text information C = (y, z) to the receiver

2) Decryption

The following steps are taken to decrypt a cipher text: The receiver needs the private key x to decrypt
The receiver picks up the cipher text information C = (y, z)
He then computes $r = y^{p-1-x} \bmod p$
The receiver finally computes $m = (r * z) \bmod p$ to extract the original message

3) Signing And Verification Algorithm

This signature process aims at signing a message to ensure message authentication and integrity.
There are two processes involved in this section, they are; Signature Generation Process Signature Verification Process.

4) Signature Generation Process

The sender should do the following
Pass the numerical representation m of the message into an hash function to produce an hashed message M (i.e. M = hashfunction(m))
Choose a secret key x such that $1 < x < p-1$
Choose random integer k with $1 <= k <= (p-1)$ and gcd(k, p-1) = 1(gcd is the grand common divisor)
Compute $h = a^k \bmod p$
Compute: $k\_Inverse = k^{-1} \bmod p$
Compute the value: $s = (M - (x * h)) * -$ k_Inverse mod (p - 1)
Compute $s = (p – 1) – s$

5) The signature is the tuple (h, s)

Signature Verification Process
The receiver should do the following
Collect the signature (h, s)
Compute $b = a^x \bmod p$
Compute $var1 = (b^h * h^s) \bmod p$
Compute $var2 = a^M \bmod p$
If var1 == var2 then signature is valid otherwise invalid.

B. RSA Algorithm

The security of RSA is inherent in the difficulty of factoring large numbers. The RSA encryption and decryption algorithms require a single modular exponentiation operation. The size of the modulus determines the security strength of the cipher [2].

i) Key generation

The algorithm is stated as follows;
Generate two large random(and distinct) primes p and q, each roughly the same size.
Compute n=p.q and $Ø = (p-1)(q-1)$.
Select a random integer e, where $1<e< Ø$, such that the greatest common divisor, gcd(e, Ø)=1.

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Use Extended Euclidean algorithm to compute the unique integer where $1<d< Ø$, such that $e \equiv 1 \pmod{Ø}$.
Sender's public key is (n,e) and private key is d.

ii) Encryption and Decryption

m is the message

1) Encryption
   This is done using the public key, (n,e)
   $c = m^e \bmod n$.
2) Decryption
   This is done using the private key, (d,n)
   $m = c^d \bmod n$.
3) Signing and Verification
   This has two stages which are the signature generation and signature verification.
4) Signature Generation
   The Sender should do the following:
   Compute m = R(m), an integer in the range [0,n-1].
   Compute $s = m^d \bmod n$.
   Sender's signature form is s.
5) Signature verification
   To verify Sender's signature
   Compute var1= m mod n
   Compute $var2 = s^e \bmod n$
   If var1 = var2, Signature valid else invalid

V. RESULTS

The results obtained are shown below as figures and tables.

Table 1: Execution Times for Encryption and Signing

Text Length in characters	ElGamal	RSA
18580	29083.6349ms milliseconds (29s)	3818.8579milI(3s)
9242	13232.2388 milli (13s)	641.4635milli(0s)
6095	8380.3855milli (8s)	224.8633milli(0s)
4680	6502.8839milliseconds (6s)	142.941milli(0s)
3739	5199.6898milliseconds (5s)	108.6065milli(0s)
3209	4462.4788milliseconds (4s)	83.7358milli(0s)
2762	3731.9332milliseconds (3s)	67.6062milli(0s)
2524	3490.1043milliseconds (3s)	62.7788milli(0s)
2247	3115.9792milliseconds (3s)	56.3892milli(0s)
2083	2892.6267milliseconds (2s)	49.9453milli(0s)

Table 2: Execution Times for Decryption

Text Length in characters	ElGamal	RSA
18580	111.9454milliseconds (0s)	162.4227milliseconds (0s)
9242	40.795milliseconds (0s)	63.6866milliseconds (0s)
6095	23.8803milliseconds (0s)	40.5696milliseconds (0s)
4680	19.4903milliseconds (0s)	32.777milliseconds (0s)
3739	16.9437milliseconds (0s)	28.0812milliseconds (0s)
3209	13.2764milliseconds (0s)	23.5664milliseconds (0s)
2762	12.1137milliseconds (0s)	19.2534milliseconds (0s)
2524	10.3053milliseconds (0s)	16.4246milliseconds (0s)
2247	9.4394milliseconds (0s)	14.9514milliseconds (0s)
2083	8.5182milliseconds (0s)	14.3429milliseconds (0s)

Table 3: Execution Times for Signature verification

Text Length in characters	ElGamal	RSA
18580	3803.3193milliseconds (3s)	2013.0053milliseconds (2s)
9242	1216.9236milliseconds (1s)	332.1506milliseconds (0s)
6095	705.5812milliseconds (0s)	133.1629milli(0s)
4680	526.0838milliseconds (0s)	94.382milli(0s)
3739	413.8463milliseconds (0s)	61.7007milli(0s)
3209	348.0134milliseconds (0s)	50.902milli(0s)
2762	302.8908milliseconds (0s)	39.5771milli(0s)
2524	277.0682milliseconds (0s)	32.6042milli(0s)
2247	242.8458milliseconds (0s)	27.1777milli (0s)
2083	224.3202milliseconds (0s)	27.0507milli(0s)

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

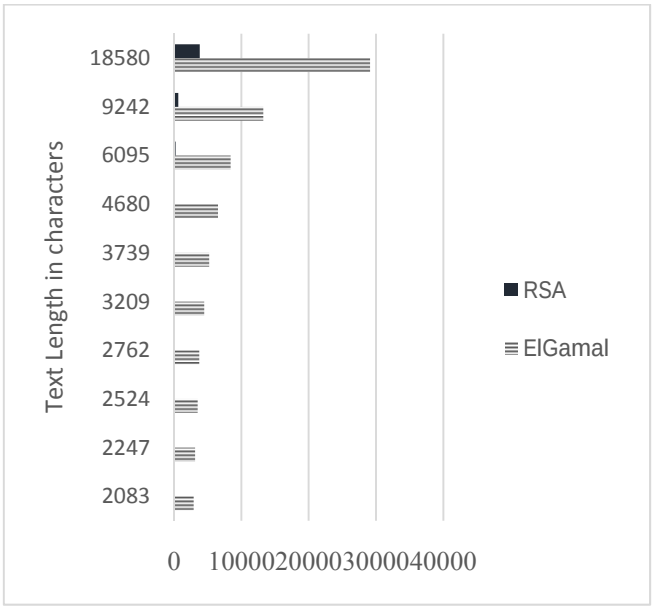

Fig 1: Execution Times for Encryption and Signing

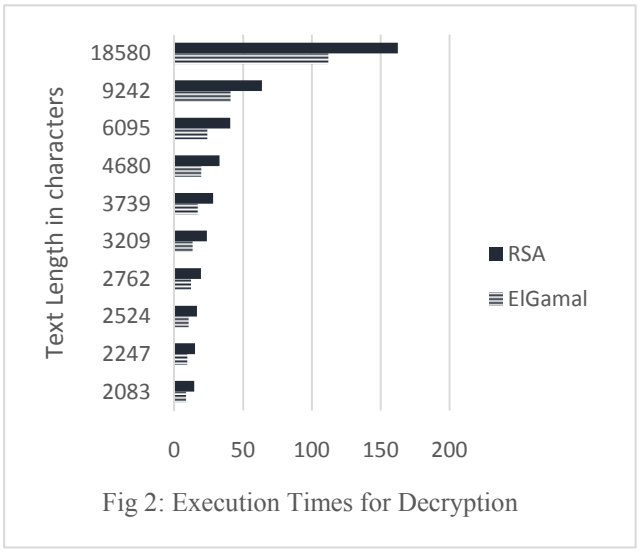

Fig 2: Execution Times for Decryption

Fig 2: Execution Times for Decryption

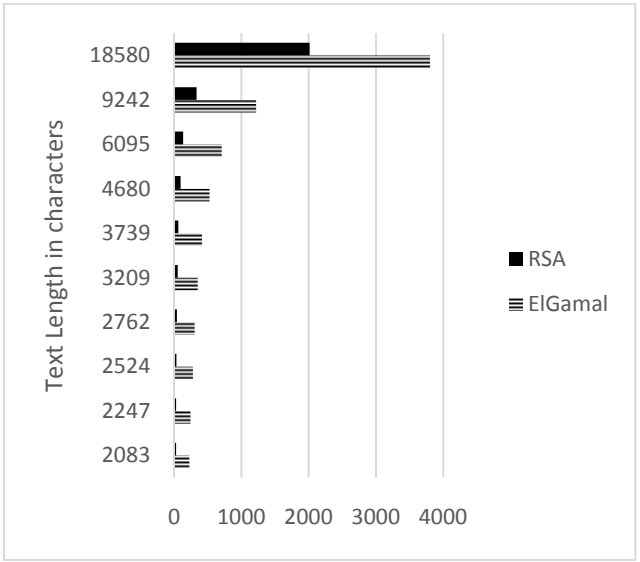

Fig 3: Execution Times for Signature verification

VI. ANALYSIS OF RESULTS

The Execution times for both the Elgamal and RSA algorithms are shown on the Tables and Figures. The times are measured in milliseconds, but converted to seconds as displayed on the result templates. We observe and deduce as follows from the results obtained

In the encryption and signing process, the RSA performs better than Elgamal in all cases.In the decryption process, the Elgamal outperforms RSA; meaning that text messages are decrypted faster by Elgamal than does the RSA technique. In the signature verification process, the RSA again performs better than the Elgamal approach.

When viewed as a single tool, the RSA is superior to the Elgamal algorithm in terms of computational speeds. This, in part, explains why the RSA algorithm has been and is still being used in designing many security protocols for data communication.

VII. CONCLUSION

From this study, we have observed that even though the RSA is superior to the Elgamal on the overall assessment, it is not as efficient as Elgamal when the rate of data decryption is considered. It is therefore fathomable that a platform that will hybridize both approaches may yield a more efficient technique than either the Elgamal or RSA algorithm. Hence efforts at designing a hybrid algorithm of these two techniques are strongly recommended as candidates for further research work. Furthermore, other performance evaluation parameters apart from energy and speeds may be investigated. Measures such Halstead, Cyclomatic, Lines-of-code and related ones could be computed, to enable us conclude with greater probability which of RSA and Elgamal algorithms is more efficient for pragmatic purposes.

ACKNOWLEDGEMENT

The efforts of Oyewole Samson Opeoluwapo in computer programming are greatly recognized and appreciated.

REFERENCES

[1] Adam J. Elbirt (2008); Understanding and Applying Cryptography and Data Security, Auerbach Publications, Taylor and Francis Group.

[2] Padmavathi, D.Shanmuga Priya (2009); “A Survey of Attacks, Mechanisms and challenges in wireless, Sensor Networks” International Journal Of Computer Science and Information Security, Vol.4, No. 1&2.

[3] Hans Delfs, Helmut Knebl (2007); Introduction to Cryptography; Springer-Verlag Berlin Heidelberg.

[4] R. Rivest (1992); The MD5 Message-Digest Algorithm Internet Request for Comments; Presented by Rump session of Advances in Cryptography – CRYPTO’ 91.

[5] T. El Gamal (1985); A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. IEEE Transactions on Information Theory 31(4):469 472.

[6] Xiahua Luo, Kuogen Zheng (2004); “Encryption algorithms comparisons for wireless networked sensors”; IEEE International Conference on Systems, Man and Cybernetics, College of Computer Science, Zhejiang University, China.