# Digital Signature with Multiple Signatories Based on Modified ElGamal Cryptosystem

Aris J. Ordonez
*Graduate Programs*
*Technological Institute of the Philippines*
Quezon City, Philippines
sirarisordonez@gmail.com

Bobby D. Gerardo
*West Visayas State University*
Iloilo, Philippines
bobby.gerardo@gmail.com

Ruji P. Medina
*Graduate Programs*
*Technological Institute of the Philippines*
Quezon City, Philippines
ruji.medina@tip.edu.ph

***Abstract*** **– El Gamal Digital Signature typically works between one signatory and one verifier. This scheme will not work in situations where there are two (2) or more signatories. This paper proposes a variant of the El Gamal Digital Signature Scheme that allows multiple signatories in a single signature by modifying the ElGamal Cryptosystem. The modification allowed multiple senders to encrypt a plaintext message into a ciphertext using multiple private keys. The provision also provided a scheme for a digital signature that allowed multiple signatories be tendered in a document in a single signature.**

***Keywords—ElGamal Encryption; Digital Signature, Multiple Signatories; Information Security; Cryptography***

## I. INTRODUCTION

Digital signatures are merely an implementation of cryptography that simulates the function of a real signature. Signatures normally are manifestations of its owner's approval of specific contentions, agreement of provisions, receipt of an entity, or simple notification. It is also a legal basis for an individual's agreement in written manifests and contracts. However, as the society shifted to digital technology, the implementation of signatures also turned digital.

The Digital Signature Standards (DSS) set the standards for the utilization of digital signatures [1]. Hash functions and in combination with the cryptographic scheme are the standard methods for the implementation of this scheme. Authenticity, Integrity, and Non-repudiation [2] are the other key elements considered by the DSS in setting the standards.

There are lots of digital signature schemes that are accepted by the DSS. Among these are the RSA signature [3], Lamport Signature [4], Merkle signature [5], Rabin signature [6], and the ElGamal Signature [7]. All of these digitals signatures are anchored on modern cryptography that utilizes private and public key for security. All these, however, are basically designed to handle one signatory and one verifier at a time only.

Aside from cryptography, another essential element of a digital signature is the hash function. A hash function is a method that converts data (regardless of its size) into a fixed length string called hash code. It also has the capability to convert back the hash code into its original form.

The process of the digital signature scheme starts with subjecting the document to be signed in a Hash Function in order to produce a fixed-length hash code. The hash code is encrypted using standard cryptography. The ciphertext derived from the process became the digital signature. Together with the original document, the digital signature is sent to the verifier for checking [8].

To check for the signature's authenticity, integrity, and non-repudiation, the verifier hash the original document using the hash function used in producing the digital signature to produce its hash code. The digital signature is then decrypted to produce its plaintext form which happens to be the document's hash code also. The results are compared. Any slight discrepancy between the two hash codes compromises either, the authenticity of the signature, or the integrity of the document. Similar hash codes mean authentic signature was used, the document is intact, and the signatory has no way of denying that he tendered the signature (non-repudiation).

Multiple digital signature schemes are merely an extension of the existing digital signature schemes. These are modified digital signature schemes that allow multiple signatories in a single provision or document [9]. Multiple digital signatures are necessary for situations where multiple signatories are required to be tendered in a single document.

There are several ways of doing multiple digital signatures. One is the sequential method [9] where a document to be signed is hashed and encrypted to produce the first signature. The resulting signature is signed again using a similar process to produce a different signature. The same process is done repeatedly depending on the number of signatories required to sign the document. This method produces a single signature that underwent multiple signing processes. In order to verify the signature, the signature is decrypted opposite of the process that was used during its encryption.

Parallel Multiple Digital Signature [9] is merely a duplication of the usual digital signing process. A document is hashed and encrypted using the signatory's private key to produce a digital signature. To create the second signature, the same document is hashed again and encrypted using the private key of the second signatory. This is repeatedly done depending on the number of signatories required by the process. What is

978-1-5386-5254-1/18/$31.00 ©2018 IEEE

sent to the verifier are the original document and the different signatures produced by the process.

The sequential signing process has a drawback. The method requires that the process is conducted in a series one after the other. It will only work in a situation where a hierarchy of signatories exists. The parallel process, on the other hand, does not produce a single signature but rather produces multiple signatures that are outputs of separate sessions. It is merely a duplication of the existing digital signature schemes. What this paper accomplished is a method that allows multiple signatories in a single signature that is simultaneously signed. The method, unlike the parallel signature scheme, only produces a single digital signature. The method provided an extra layer of protection to the system due to the additional number of signatories and added complexity to the process because of the additional number of private keys used in the process.

## II. The Underlying Cryptosystem

A cryptosystem is a method intended to protect classified information that will pass through the unsecured channel. The intention of the process is to make it possible that information may only be accessed by the intended recipient regardless of its method of delivery. The concept is to convert a plaintext message into an incomprehensible form using a key. Only the intended recipient may convert the encrypted message into its original form using his own private key.

Normally the same key is used in the encryption and decryption process. Compromising the key would also compromise the message that is why the key exchange should be done covertly. The Diffie-Hellman Key Exchange Algorithm [10] provides a method of exchanging keys through public domain without compromising its value. This was made possible through the utilization of the Discrete Logarithm Problem (DLP)[11] which is a concept used in most cryptosystems that utilizes a mathematical operation involving modular arithmetic.

The concept is for the receiver and sender to keep private keys among themselves. Create a public key through a computation using public parameters and share the derived public key and parameters in public but keep the private keys among themselves. The message will be encrypted using these keys and will be channeled into a public domain. Although the public may be able to see the encrypted message together with the public keys and public parameters, it would be impossible for them to decrypt the message without the acquiring private keys.

### *A. El Gamal Cryptosystem*

The ElGamal Cryptosystem [12] is an encryption algorithm that is based on discrete logarithm problem [11] and the Diffie-Hellman Key Exchange Algorithm [10]. It has three phases: The Key Generation Phase, Message Encryption Phase, and the Decryption Phase. The method requires both the sender and the receiver to each provide their own private keys which they will keep among themselves. A public key will be generated and exchanged by both parties from the private keys using the Diffie-Hellman Key Exchange Algorithm. The plaintext message will be encrypted using the sender's private key and the publicly shared keys. The message may be decrypted by the receiver using his own private key and the publicly shared keys. The El Gamal Cryptosystem, however, only works between a single sender and a single receiver.

#### *1. Key Generation Phase*

The Key Generation Phase generates significant keys necessary for the encryption and decryption process of the cryptosystem. It starts with the generation of $P(prime)$ and a cyclic group G = $\{1, . . ., P-1\}$. A very large prime number is ideal to ensure that the implementation of the DLP would be hard. A generator $g$, the sender's private key $x$ and the receiver's private key $a$ are also chosen from $G$. From these values:

$$b = g^a \ mod \ P \tag{1}$$

is computed by the intended recipient.

The receiver's private key is $a$ and the public key would be $\{g, P, b\}$.

#### *2. Message Encryption Phase*

Encryption starts with the sender's identifying a random number $x$ from the cyclic group. Knowing the public keys $\{g,P,b\}$ and the message $m$, also from the cyclic group $G$, the sender computes for $\{c_1,c_2\}$ *where:*

$$c_1 = g^x \ mod \ P \tag{2}$$
$$c_2 = m \cdot b^x \ mod \ P \tag{3}$$

The encrypted message is $\{c_1,c_2\}$ which contains the public key which together with the private key is necessary for decoding, and the encrypted data which will be sent to the intended recipient for decryption.

#### *3. Decryption Phase*

Upon receiving the encrypted message $\{c_1,c_2\}$, the receiver may decrypt the message $m$ by computing:

$$m = c_2 \,/\, c_1^a \ mod \ P \tag{4}$$

Where $a$ is the receiver's private key, $P$ is from the public key, and $m$ is the message.

### *B. The Modified El Gamal Cryptosystem*

The ElGamal Cryptosystem in its basic form would not allow a process that would allow multiple senders to a single recipient in an encryption process. A modification is necessary in order to allow such process and to eventually make the algorithm applicable to a digital signature scheme requiring multiple signatories.

Similar to the ElGamal Cryptosystem, the process starts with the Key Generation Phase wherein the receiver provides a public key $b$ generated from its own private key $a$. A large prime number $P$ then is determined together with the generator $g$ from the cyclic group $G = \{1 \ ... \ P-1\}$. The private key of the sender is also selected from the cyclic group $G$. The prime number $P$ should be a very large number to ensure that the

DLP which is the core of the key exchange process is hard to break. The value of *b* is computed using the formula:

$$b = g^{a} \bmod P \qquad (5)$$

The public key *{g, P, b}* is shared publicly and *a* is kept in private by the receiver.

With the availability of the public keys *{g, P, b}*, *n* number of senders encrypt the plaintext message *m* using their individual random private keys $\{R_1, R_2, R_3, \ldots R_n\}$. The values for $c_{11}$, $c_{12}$, $c_{13}$, $c_{14}$, ….. $c_{1n}$ and the value of $c_2$ are then computed using the following formula:

$$c_{11} = g^{R}_{1} \bmod P \qquad (6)$$
$$c_{12} = g^{R}_{2} \bmod P \qquad (7)$$
$$c_{13} = g^{R}_{3} \bmod P \qquad (8)$$
$$c_{14} = g^{R}_{4} \bmod P \qquad (9)$$
$$c_{15} = g^{R}_{5} \bmod P \qquad (10)$$
$$\vdots$$
$$c_{1n} = g^{R}_{n} \bmod P \qquad (11)$$

*and*

$$c_2 = \frac{(m \cdot b^{R}_{2} \cdot b^{R}_{4} \cdot \ldots b^{R}_{u} \cdot)}{(b^{R}_{1} \cdot b^{R}_{3} \cdot \ldots b^{R}_{v})} \bmod P \qquad (12)$$

where:

*n = senders*
*u = {x|x mod 2=0 and x<=n}*
*v = {x|x mod 2=1 and x<=n)}*

The generated encrypted message then would be in the form of $\{c_{11},c_{12},c_{13},c_{14},\ldots,c_{1n},c_2\}$ which will then be sent to the intended recipient.

To decrypt the message *m*, the receiver computes the value of *m* using the formula:

$$m = \frac{(c_2 \cdot c_{11}^{a} \cdot c_{13}^{a} \cdot \ldots c_{1v}^{a})}{(c_{12}^{a} \cdot c_{14}^{a} \cdot \ldots c_{1u}^{a})} \bmod P \qquad (13)$$

where:

*u = {x|x mod 2=0 and x<=n}*
*v = {x|x mod 2=1 and x<=n}*
*n = senders,*
*a = receiver's private key, and*
*m = plaintext message.*

## III. SIMULATION OF MODIFIED EL GAMAL CRYPTOSYSTEM

### *A. Five Senders and Single receiver*

The example below simulates the process using five (5) senders and one (1) receiver. The characters for cryptography, Bob (receiver) and Alice (sender) will be used for this simulation.

1. Key Generation Phase

Let the generator *g = 48*, and prime number *P = 97*. Bob's (receiver) private key is *a = 36*. Following Formula 5, Bob computes for:

$$\begin{aligned} b &= 48^{29} \bmod 97 \\ &= 22 \end{aligned}$$

Bob publishes the values {g=48, P=97, b=22} as the public keys.

2. Encryption Phase

Alice₁, Alice₂, Alice₃, Alice₄, and Alice₅ (senders) decides on their random private keys. In this example they chose $R_1=30$, $R_2=25$, $R_3=12$, $R_4=42$, *and* $R_5=10$ respectively. They need to encrypt *m = 47* and send it to Bob as an encrypted message. Following Formula 6 to 11:

Alice₁ computes for:

$$\begin{aligned} c_{11} &= 48^{30} \bmod 97 \\ &= 50 \end{aligned}$$

Alice₂ computes for:

$$\begin{aligned} c_{12} &= 48^{25} \bmod 97 \\ &= 49 \end{aligned}$$

Alice₃ computes for:

$$\begin{aligned} c_{13} &= 48^{12} \bmod 97 \\ &= 75 \end{aligned}$$

Alice₄ computes for:

$$\begin{aligned} c_{14} &= 48^{42} \bmod 97 \\ &= 64 \end{aligned}$$

Alice₅ computes for:

$$\begin{aligned} c_{15} &= 48^{10} \bmod 97 \\ &= 9 \end{aligned}$$

Simultaneously, the senders compute the value of $c_2$ following Formula 12.

$$\begin{aligned} c2 &= (47 \cdot 22^{25} \cdot 22^{42})/(22^{30} \cdot 22^{12} \cdot 22^{10}) \bmod 97 \\ &=33 \end{aligned}$$

The encrypted message would be in the form of $\{c_{11},c_{12},c_{13},c_{14},c_{15},c_2\} = \{50,49,75,64,9,33\}$ which will be sent to Bob for decryption.

3. Decryption Phase

Upon receiving the encrypted message $\{c_{11},c_{12},c_{13}, c_{14},c_{15},c_2\} = \{50,49,75,64,9,33\}$ from Alice₁, Alice₂, Alice₃, Alice₄,and Alice₅, Bob extracts the value of *m* (message) following Formula 13.

$$\begin{aligned} m &= (13 \cdot 50^{36} \cdot 75^{36} \cdot 9^{36}) / (49^{36} \cdot 64^{36}) \bmod 97 \\ &= 47 \end{aligned}$$

The value of *m* which is *47* is the plaintext message which was encrypted by the five (5) Alices.

### *B. Six Senders and Single Receiver*

1. Key Generation Phase

Let the generator $g = 68$, and prime number $P = 101$. Bob, the receiver's private key is $a = 89$. Following Formula 5, Bob computes for:

$$\begin{aligned} b &= 68^{89} \bmod 101 \\ &= 25 \end{aligned}$$

Bob publishes the values {g=68, P=101, b=25} as the public keys.

2. Encryption Phase

The senders Alice$_1$, Alice$_2$, Alice$_3$, Alice$_4$, Alice$_5$, and Alice$_6$ decides on their random private keys which are $R_1=77$, $R_2=23$, $R_3=89$, $R_4=56$, $R_5=5$, and $R_6=34$ respectively. They need to encrypt $m = 66$ and send it to Bob as an encrypted message. Following Formula 6 to 11:

Alice$_1$ computes for c$_{11}$

$$\begin{aligned} c_{11} &= 68^{77} \bmod 101 \\ &= 79 \end{aligned}$$

Alice$_2$ computes for $c_{12}$

$$\begin{aligned} c_{12} &= 68^{23} \bmod 101 \\ &= 78 \end{aligned}$$

Alice$_3$ computes for $c_{13}$

$$\begin{aligned} c_{13} &= 68^{89} \bmod 101 \\ &= 25 \end{aligned}$$

Alice$_4$ computes for $c_{14}$

$$\begin{aligned} c_{14} &= 68^{56} \bmod 101 \\ &= 58 \end{aligned}$$

Alice$_5$ computes for $c_{15}$

$$\begin{aligned} c_{15} &= 68^{5} \bmod 101 \\ &= 87 \end{aligned}$$

Alice$_6$ computes for $c_{16}$

$$\begin{aligned} c_{16} &= 68^{34} \bmod 101 \\ &= 92 \end{aligned}$$

Simultaneously, the senders compute for the value of $c_2$ using Formula 12.

$$\begin{aligned} c2 &= (66 \cdot 25^{23} \cdot 25^{56} \cdot 25^{34})/(25^{77} \cdot 25^{89} \cdot 25^{5}) \bmod 101 \\ &=51 \end{aligned}$$

The encrypted message would be in the form of $\{c_{11},c_{12},c_{13},c_{14},c_{15},c_{16},c_2\} = \{79,78,25,58,87,92,51\}$ which will be sent to Bob.

3. Decryption Phase

Upon receiving the encrypted message $\{c_{11},c_{12},c_{13}, c_{14},c_{15}, c_{16},c_2\} = \{79,78,25,58,87,92,51\}$ from Alice$_1$, Alice$_2$, Alice$_3$, Alice$_4$, Alice$_5$, and Alice$_6$, Bob extracts the value of $m$ (message) using Formula 13:

$$\begin{aligned} m &= (51 \cdot 79^{89} \cdot 25^{89} \cdot 87^{89}) / (78^{89} \cdot 58^{89} \cdot 92^{89}) \bmod 101 \\ &= 66 \end{aligned}$$

The plaintext message is 66.

The examples provided, are the underlying cryptosystem that served as the foundation of the proposed digital signature scheme. With the multiple senders provided in the system, the method also opened an opportunity to also use the process in a digital signature scheme that would allow multiple signatories in a single signature.

## IV. Proposed Multiple Digital Signature Scheme

Multiple Digital Signatures are merely variants of existing digital signature schemes that involve multiple signatories in the process. In most cases, the methods are merely extensions of the existing single signatory digital signatures. Fig. 1 showed how sequential multiple digital signatures work. The process starts with the document to be signed. The document is hashed using the standard hash function in order to produce its hash code which is a fixed-length string. The hash code is encrypted using public-private key cryptosystem to create a ciphertext which eventually becomes the digital signature. If multiple signatories are required, the digital signature is encrypted again to produce a different signature using the next signatory's key. The process is repeatedly done depending on the number of signatories required to sign the document. Only one signature is produced by this process.

To check for the authenticity of the digital signature, the signature is decrypted using the private keys that were used to decrypt it starting from the last one to the first. The method is done in a reverse process. Upon producing the hash code through the decryption process, the original document is also hashed to produce its original hash code that will be compared with the decrypted code. Similar hash codes mean authentic signature and uncompromised document.

This digital signing method provides an extra level of security to situations where multiple approvals are required. Another remarkable thing about the process is that it only provides a single signature regardless of the number of signatories that were associated with the process.

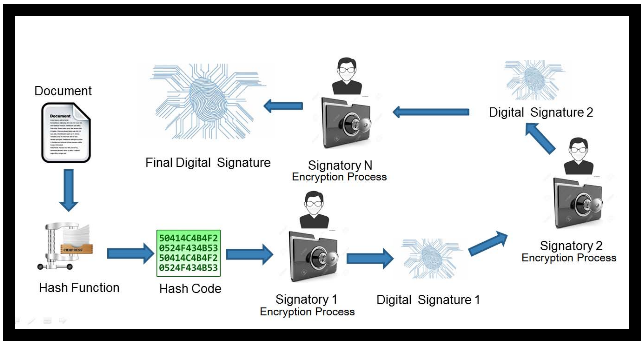

Fig. 1. Sequential Multiple Digital Signature Scheme

One of the major drawbacks of the Sequential Multiple Digital Signature Scheme is the requirement that the process of signing should be done on a specific order and its checking has to be on the exact opposite order of the signing process. The method will not be applicable in situations where the signatories need not sign in a specific sequence. It is mandatory that the signing process should be done in a specific order in order to maintain the integrity of the process.

The Parallel Multiple Digital Signature as shown in Fig. 2 is merely a duplication of the usual digital signing process. The hash code of the document that must be signed is encrypted using separate processes depending on the number of signatories required. The process produces multiple signatures equivalent to the number of signatories required to sign the document.

The disadvantage of using the Parallel Multiple Digital Signature Scheme is that the number of signatures produced by the process is equivalent to the number of signatories also. As the number of signatory increases, the number of signatures produced also increases. In some situation, the multiple signatures produced are not significant and in some cases are irrelevant. There are however situations where a single signature that represents multiple signatories is required. This is prevalent in situations where security is always an issue. The single-signature scheme with multiple-signatories in it provides added confusions to those who would attempt to break the system for their own vested interest.

The Sequential and Parallel Multiple Digital Signature Scheme is merely an extension of the cryptographic algorithm used in the system. No modification of the encryption algorithm is necessary in order to perform these methods. The processes are however modified in order to accommodate the required number of signatories if these methods will be used in situations where multiple signatories are required in a single signature.

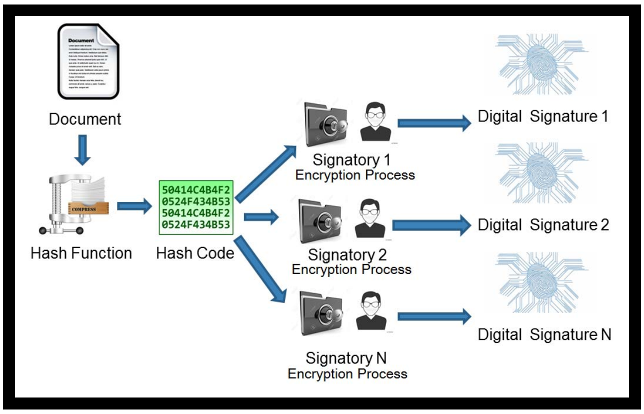

Fig. 2. Parallel Multiple Digital Signature Scheme

The proposed multiple digital signature scheme produces a single digital signature from multiple signatories through a parallel process. The signatories simultaneously tender their signatures to produce a single signature. Their private keys are merged together by the algorithm (ElGamal) during the encryption process.

Fig. 3 shows how the process is carried out. The document is hashed to produce a fixed-length hash code. The hash code is encrypted using the modified ElGamal Algorithm that requires multiple private keys (signatures) coming from the signatories. The encrypted hash code that has been produced is the digital signature of the document.

The modified ElGamal Cryptosystem presented in this paper has the capability to carry out the encryption process required by the digital signature scheme. The modified ElGamal Algorithm utilizes multiple senders with multiple private keys in the process. The provision would allow multiple signatories to be integrated into the process when applied in a digital signature scheme.

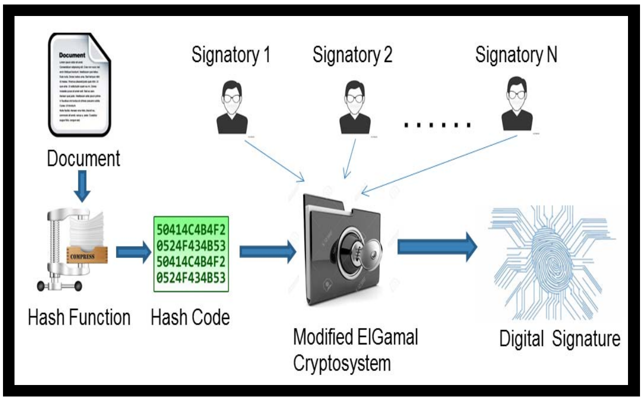

Fig. 3. Proposed Multiple Digital Signature Scheme

The decryption process of the Modified ElGamal Algorithm also provides a capability to the verifier to check for the authenticity of the digital signature. The complexity of the ElGamal specifically its utilization of the DLP provides a notion that only the individual who has knowledge of the private key could decrypt the ciphertext.

## V. Evaluation and Applications

The DSS defined the standard for implementing Digital Signature and recommends Cryptographic algorithm that may be used in connection with this. However, DSS does not limit the use of the cryptographic algorithm to specific algorithms only. As long as the scheme follows the standard set by DSS, it could use any appropriate cryptographic algorithm in line with it.

This paper tested the Modified ElGamal Algorithm which served as the foundation of the proposed Multiple Digital Signature Scheme. A test code written in Python was used in order to test the accuracy of the algorithm when subjected to a different number of inputs (senders). The test covers encryption and decryption process using an increasing number of senders and random parameters. The accuracy of the algorithm was tested by comparing the plaintext message with the decrypted ciphertext which was subjected to the modified ElGamal Cryptosystem. The result of the test was recorded in Table I.

Table I showed the different tests conducted on the modified algorithm. Different Parameters were injected including the Prime Modulo *(P),* Generators *(g)* and the Private Keys of the Receiver *(b).* The test results proved the accuracy of the cryptographic algorithm used in the proposed digital signature. This is regardless of the prime numbers used and the parameters involved in the process. There is however significant and obvious increase in the time consumed by the algorithm as the number of senders or the value of the Prime Modulo increases. This can be attributed to the added iterations required by the process which is directly proportional to the number of senders involved in the method.

TABLE I. Test Result of the Modified El Gamal Cryptosystem using Different Number of Senders, Prime Numbers, and Random Parameters

| No. of Senders | Prime Number (P) | Parameters {g,b} | Encryption (Seconds) | Decryption (Seconds) | Result |
|---|---|---|---|---|---|
| 10 | 61333 | 58766 / 57593 | 1.052104 | 1.833937 | Accurate |
| 11 | 60527 | 38529 / 19496 | 1.242910 | 0.516794 | Accurate |
| 20 | 6679 | 6075 / 5372 | 0.063786 | 0.103146 | Accurate |
| 21 | 8221 | 118 / 7017 | 0.51251 | 0.206806 | Accurate |
| 30 | 13309 | 10290 / 11540 | 0.335512 | 0.911392 | Accurate |
| 31 | 10177 | 6496 / 4177 | 0.249643 | 0.213330 | Accurate |
| 40 | 16673 | 4441 / 6564 | 1.187440 | 0.933433 | Accurate |
| 41 | 14633 | 11918 / 9970 | 1.242646 | 1.511835 | Accurate |
| 50 | 34303 | 26192 / 16621 | 6.570648 | 6.293233 | Accurate |
| 51 | 33247 | 24589 / 21968 | 6.612416 | 9.933385 | Accurate |
| 99 | 1213 | 978 / 601 | 0.077120 | 0.056585 | Accurate |
| 100 | 727 | 387 / 486 | 0.018039 | 0.031536 | Accurate |

The digital signature will be the trend of the future. The proposed method would be a replacement to the demand for digital signatures requiring multiple signatories which have to be signed simultaneously. The proposed multiple-digital signature scheme provided an extra level of protection to the underlying cryptosystem that was utilized by the scheme.

Another possible application that this method may be used will be in signing digital files where there are two or more signatories required by the process. It may also be used in some security protocols where multiple verifications are required in order to grant somebody an access to a protected system.

## VI. Conclusions

The modified ElGamal Cryptosystem provided a method that allowed the use of multiple private keys in encrypting plaintext message into a ciphertext. The method provided a way for implementing digital signatures with multiple signatories which require simultaneous signing. The proposed digital signature scheme provided a method that allowed the signatures of multiple signatories to be embedded in a single signature which is significant to some situations especially where an extra level of security is required provided by multiple monitoring and control.

The test conducted focused merely on the accuracy of the modified algorithm for digital signature. However, issues regarding the speed of the process were also observed. This can still be improved by optimizing the code implementation of the algorithm when used in an application or by utilizing tested codes for handling very large prime numbers in order to improve the speed of the process.

## References

[1] National Institute of Standards and Technology. (2009). Digital Signature Standard (DSS). *Fips Pub 186-4*, (July), 1–119. https://doi.org/10.6028/NIST.FIPS.186-4

[2] Gupta, R., Singh, S., & Maan, P. (2014). Digital Signatures – Single and Multiple On Single and Multiple Documents, *3*(3), 566–570.

[3] Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, *21*(2), 120–126. https://doi.org/10.1145/359340.359342

[4] Lamport, L. (1979). Constructing Digital Signatures from a One-Way Function. *SRI International Computer Science Laboratory*, *94025*(October), 1–8. Retrieved from http://research.microsoft.com/en-us/um/people/lamport/pubs/dig-sig.pdf

[5] G. Becker. "Merkle Signature Schemes, Merkle Trees and Their Cryptanalysis", seminar 'Post Quantum Cryptology' at the Ruhr-University Bochum, Germany

[6] Rabin, M. O. (1979). Digitalized Signatures and Public-Key Functions as Intractable as Factorization.

[7] Elgamal, T. (1985). A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, *31*(4), 469–472.

[8] Hwang, M.-S., & Lee, C.-C. (2005). Research Issues and Challenges for Multiple Digital Signatures. *International Journal of Network Security*, *1*(1), 1–7. Retrieved from http://isrc.nchu.edu.tw/ijns/

[9] Gupta, R., Singh, S., & Maan, P. (2014). Digital Signatures – Single and Multiple On Single and Multiple Documents, *3*(3), 566–570.

[10] Diffie, W., Diffie, W., & Hellman, M. E. (1976). New Directions in Cryptography. IEEE Transactions on Information Theory, 22(6), 644–654. https://doi.org/10.1109/TIT.1976.1055638

[11] A. Odlyzko, "Discrete Logarithm in finite fields and their cryptographic significance", Proc. Eurocrypt 84. to Appear.

[12] Elgamal, T. (1985). A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. IEEE Transactions on Information Theory, 31(4), 469–472.

[13] Flonta, S., & Miclea, L. (2008). An extension of the El Gamal encryption algorithm. 2008 IEEE International Conference on Automation, Quality and Testing, Robotics, AQTR 2008 - THETA 16th Edition - Proceedings, 3, 444–446. https://doi.org/10.1109/AQTR.2008.4588960

[14] Rao, F.-Y. (2017). On the Security of a Variant of ElGamal Encryption Scheme. IEEE Transactions on Dependable and Secure Computing, 1–1. https://doi.org/10.1109/TDSC.2017.2707085

[15] Hung-Min Sun and Tsonelih Hwang (1991), An Efficient Probabilistic Public-Key Block Encryption and Signature Scheme Based on El-Gamal's Scheme

[16] Mikhail, M., Abouelseoud, Y., & Elkobrosy, G. (2014). Extension and application of El-Gamal encryption scheme. 2014 World Congress on Computer Applications and Information Systems, WCCAIS 2014. https://doi.org/10.1109/WCCAIS.2014.6916627

[17] Tiersma, H. J. (1997). Enhancing the security of El Gamal â€TM s signature scheme.

[18] Harn. (1995). Enhancing the security of ElGamal's signature scheme. : IEE Proceedings - Computers and Digital Techniques, 142(5), 1995. https://doi.org/10.1049/ip-cdt:19952125

[19] He, J., & Kiesler, T. (1994). Enhancing the security of El Gamal â€TM s signature scheme, 141(4), 249–252.