REDUCED TIME COMPLEXITY VARIANT OF DIGITAL SIGNATURE ALGORITHM

[1]HAMZA A. A. AL-SEWADI, [2]MOHAMMED M. RIFAAT

Faculty of Information Technology, Middle East University, Amman, Jordan

E-mail: [1]hsewadi@meu.edu.jo, [2]mohmos07@googlemail.com

ABSTRACT

Signing and verification time of messages is an essential issue. The standard Digital signature algorithm (DSA) approved by the national institute of standard and technology (NIST) has been widely used for message authentication. Besides, many variants of DSA has been suggested in order to improve the performance, such as Yen-Laih, GOST, McCurley, and other algorithms. This paper present a modified version of the standard NISI-DSA. It incorporates the same parameters used in NISI-DSA, but it is characterized by less computational complexity and improved signature verification time which is crucial for some sensitive applications. This variant achieved an increase in the verification speed of about 100% over most DSA variants, hence reducing the signature validation time considerably, however, no improvement was noticed in the signing speed. This signature validation speed gain was achieved with the same level of security as the parameters of standard DSA were used. The algorithm is first tested with prime numbers of 20, 40, 80, and 100 bits lengths, then it is experimented with using prime numbers provided by NIST with length of 1024 bits and 128 bits for *p* and *q*, respectively. The speed of verification results shows more than two times faster than NIST-DSA.

Keywords: *Digital Signature Algorithms, Time Complexity, Cryptography, Discrete Algorithms, Message Authentication, Data Integrity.*

1. INTRODUCTION

Since the ancient times there were uncounted tries to secure and protect information using different methods like Alphabet shift ciphers, or converting plaintext into unintelligible form (ciphertext) during transmission to be recovered at the receiver side. All those methods intentions are to hide the data from the public and keeps it readable within specific parties. This issue is referred to by the term “cryptography”, which is a Greek word meaning “hidden writing”. Cryptography continued development during centuries till today, and a better definition is given, i.e. “Cryptography is the study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, entity authentication, data origin authentication” [1], non-repudiation, and e-government systems” [2]. It is also used in copyright and ownership protection of various documents, such as its use in pdf files of Adobe Company [3].

Signature was used and still as assurance that the signed document is original and authentic [4]. But nowadays the world become a small village because of the communication networks which made the dream of communicating over long distances becomes true, therefore many processes can be done from far away such as money transfer, decisions making, financial business, remote election voting for elections and many other daily activities. All activities can be done over the internet today, however internet environment is not safe to most of these activates due to intruders and criminals who try to fake, steel, masquerade, make money using illegal way, etc. Such acts can ruin whole company, business, or even breach national security with faked signature or faked message. For all these actions, sender authentication and/or message integrity must be proved before any decision to be taken, therefore in this digital world, an action equivalent to hand written signature is required in order to be sure that any received signed document is genuinely signed by the real involved person, so the digital signature is the most suitable solution for authentication over the communication [4].

After the brief introduction in section 1, recent works on digital signature is listed in section 2. Then section 3 included theoretical background on the digital signature algorithm DSA and some of its variants that are closely related to the modified DSA algorithm which is proposed in section 4. Section 5 listed the obtained results of implementing the proposed algorithm with their discussion. Section 6 gives the speed gain summary of the research achievement, and section 7 concludes the paper.

2. RELATED WORK

Many digital signature algorithms such as Rivest, Shamir, and Adelman (RSA), Schnorr algorithm, ElGamal algorithm, the Standard Digital Signature Algorithm (NIST-DSA), DSA variants, Elliptic Curve Digital Signature Algorithm (ECDSA), and many others were reported in the second half of the 20th century [1], [4]. These algorithms vary in the way of mathematical operation for generating the signature and its verification which affect the speed and performance of the algorithm. DSA and its variants will be described in section 3 as they are closely related to the work in this paper. However, some recent works on digital signature are listed in the following.

Tan et.al [5] in 2003 proposed a scheme based on composite discrete logarithm. They claim their scheme is 13.5 times faster than RSA, 3 times faster than DSA, and also faster than some other scheme. It is also secure against existence forger using adaptive chosen message attack.

Poulakis, [6] in 2009 presented a modified variant of DSA algorithm based on a factorization problem. It implements two discrete logarithms and claims security strength at least equal to DSA and withstanding all known attacks. It utilizes RSA scheme properties to introduce two modular exponentiations and a modular multiplication for the signature.

Hernandes et. al. [7] in 2014 used flow analysis as a technique to describe standard network behavior, they also used genetic algorithm to optimize the process.

Singh et. al. [8] in 2015 proposed a scheme to detect any tampering and also support the image compression. Digital signature is transmitted along with the image itself, and the receiver regenerates the signature corresponding to the received image, then if both signatures match, the received image is authentic.

Tan et. al. [9] in 2016 proposed an approach to build a secure variant of rainbow (multivariate Digital Signature). They claims that their variant can resist all the existing attacks according to their security analysis, but no mention of the signing or verification speed.

Andrade and Terada [10] in 2016 presented a new digital signature algorithm for Quartz digital signature based on Hidden Field Equations with vinegar - minus trapdoor, with special choice of parameters, which can resist algebraic attack. They claim efficiency gain in signature verification using vector initialization for both signature and verification algorithms.

Dhagat and Joshi [11] in 2016 proposed a method for digital signature that uses another signer as proxy, providing the ability to the sender to delegate his signature to another signer, and claim that their scheme provides protection to proxy signer of private key. The certification authority holds identity of signer, delegate duration and imposes rules on the signing ability. The scheme uses protected nominative signature so that signer and proxy cannot deny each other.

Manickam and Kesavaraja [12] in 2016 proposed a secure multi-server authentication system, utilizing advantages of digital signature in addition to elliptic curve, they claim that their system is secure against majority of possible active attacks, besides it provide low communication cost. They claim to be more adoptable for smart cards and other less energy saving devices.

So many work is reported on different methods for digital signature algorithms, such as elliptic curve digital signature algorithm (ECDSA) and as RSA digital signature algorithm, see Al Imem [13] for example. He studied and compared these two algorithms and claims ‘they have proven their strength in facing cyber-attacks, data encrypt/decrypt speed, and their competence for data integrity.

This paper presents a modified version of the standard NIST-DSA with the aim of getting improvements in the signing and verification processes. The modification enhanced the speed of signature verification significantly. In the following some theoretical background related to the standard DSA and its variant that are found necessary to be included.

3. THEORETICAL BACKGROUND

Digital Signature is a mathematical way to sign digital documents in the digital world. It is a technique that insures both the integrity of the received digital data and the authenticity of communicating bodies or persons as detailed in [4]. Sending documents over the internet is fraught with dangers of altering, changing, masking, denial, identity theft and other types of hazards. The signed data can't be denied by the sender because of the unique way and parts it contains in the signing process (Non-repudiation), which provides authenticity. Digital Signature provides the following [14].

- ***Authentication*:** Digital signature provides verification of the signer.
- ***Integrity*:** It provides a way to protect the message from being altered.
- ***Non-repudiation*:** The signer cannot deny the signature because no one else has the signing key.

Digital signature provide ***Confidentiality*** if the message signed then encrypted with encryption algorithm, no third person can decrypt it if he/she doesn't have the keys [15]. Digital signature can be used even over non-secure communication channels as it provides the senders identity and ensures that the document (data) has not been changed or altered during transmission.

3.1 Discrete Logarithms in Cryptography

The public-key cryptography technique such as RSA is designed according to the computational difficulty of Integer Factoring Problem (IFP) solving into its large primes [16]. The Diffie-Hellman key exchange [17], El-Gamal cryptosystem [18], and some of public-key cryptosystems are designed according to another computationally difficult number theory problem, which is Discrete Logarithms Problem (DLP). The standard Digital Signature Algorithm (DSA) that is set and applied by the National Institute of Standards and Technology (NIST) is designed according to computing and solving discrete logarithm problems [19].

3.2 NIST – DSA

Digital Signature Algorithm or DSA is based on DLP and its standard version, namely NIST-DSA has been practically in use since its consideration by NIST, and is briefly descried here. Three stages are involved in the use of DSA; key generation, signing, and signature verification as follows

The DSA design starts by setting the required parameters, which are two types; public and private, as listed with their rules and limitation in table 1. Typical values for the lengths of integers p and q are selected to be 1024 bits and 160 bits, respectively, and having the property that their greater common deviser, gcd = 1. Details of the implementation of DSA to select and calculate involved parameters can be seen in NIST-FIB 186-4 [1]. The parameters p, q, g, and y are public, while integer x is private key for the message signer and k is a secret random number generated for each signed message. The public key y is calculated as defined in table 1 and used in equation 7.

Table 1: Parameters Rules and Description for NIST-DSA

Parameter	Rule
p (public)	Large prime integer $2^{L-1} < p < 2^{L}$ L= the length of bits of (p).
q (public)	Large prime integer devisor of $(p-1)$, $2^{N-1} < q < 2^{N}$, N=the length of (q).
h	Random Number Less than q, used to calculate *g*
g (public)	Subgroup generator of $(q \; mod \; p)$, $1 < g < p$, $h^{(p-1)lq} \bmod p$
x (private)	Private integer, $0 < x < q$
y (public)	The signer's public key, $y = g^{x} \bmod p$
k (private)	Secret random integer that is changed for each message (unique), $0 \le k \le q$.

Now to sign a message M, the signer calculates its hash value *H(M)* first, then determine its signature r and s using equations 1 and 2. (Note: DSA uses SHA-1 as hash algorithm)

$$r = (g^{k} \bmod p) \bmod q \quad (1)$$

$$s = (k^{-1} H(M) + x.r) \bmod q \quad (2)$$

The hash function of the message is used instead of the message itself in order to ensure its integrity and present it with unique value much shorter than the message itself. Then, the signature *r* and *s*, together with the message itself *M* are sent over to the receiver.

On the receiver side the receiver can verify the received message using the signature r and s as follows.

First the receiver calculates the hash value *H(M)* of the received message using the same hash algorithm, then verifies the signature using equations 3 to 6.

$$w = s^{-1} \bmod q \qquad (3)$$

$$u_1 = (H(M).w) \bmod q \qquad (4)$$

$$u_2 = (r.w) \bmod q \qquad (5)$$

$$v = (g^{u_1}.y^{u_2} \bmod p) \bmod q \qquad (6)$$

If the calculated value v is equal to the received *r* then the signature of the message is authentic, otherwise the signature is rejected because of alteration or modification during transmission.

The block diagram illustrating the signing and the verification process for the DSA is shown in Figure (1).

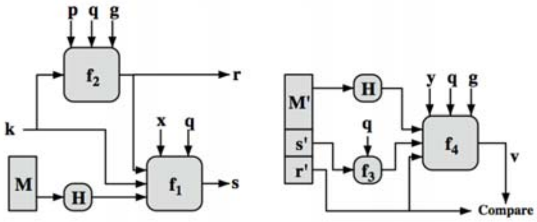

Figure 1: Block Diagram of Signing and Verification Processes Of DSA

3.3 Related DSA Variants

Due to the continuous need for faster signing and verification processes, so many variant of the standard NIST-DSA have been developed with the target to improve the performance of the digital signature, by reducing the computational complexity and decreasing processing time. Schnier [4] and Stalling [14] have listed many of these digital signature algorithms other than the standard NIST-DSA, such as the use of the pioneer protocol of Diffie-Helman for information exchange, ElGamal algorithm, RSA authentication algorithm, and Schnorr Digital Signature algorithm.

Many DSA variants were reported which aim to increase DSA efficiency and reduce the computation time. Some of DSA variants are briefly listed below.

The Russian version of digital signature algorithm, "Gosudarstvennyi Standard of Russian Federation named, GOST" [20]. It uses different equation for calculating s as will be seen in table 2 and obviously different verification equations as shown in table 3. Moreover, GOST algorithm, also differs from NIST_DSA by using a value q equals to 256 bits length instead of 160 bits.

Yen-Laih [21], uses exactly the same parameters used by NIST_DSA, however, it made the computation easier on the verification side. Likewise, McCurley [22], also makes computation easier on the verifier side and they both allow for batch verification which is not quite secure [3].

Another DSA-variant reported by Ali [23], which is also suitable for data integrity and authenticity, rather than encryption. It achieved improvement in the signature verification time using the same set of parameters used by NIST-DSA algorithm. The reduction in the time complexity was achieved by reducing the number of modular operations.

All these four variants use the parameters p, q, and same key generation process for the public key y, which is calculated by equation 7.

$$y = g^{x} \bmod p \qquad (7)$$

Then the signature value, *r* is calculated by equation 8.

$$r = (g^{k} \bmod p) \bmod q \qquad (8)$$

But the other value of the signature, i.e. s is calculated by different equation for different variants, as listed in table 2.

Table 2: Signature Modified Equations for DSA Variants

Algorithm	Signature Equation
GOST-DSA Variant [20]	$s = (x.r + k.H(M)) \bmod q$
Yen-Laih-DSA-Variant [21]	$s = ((r.k - H(M)).x^{-1}) \bmod q$
McCurley-DSA Variant [22]	$s = (k.H(M) + x.r)^{-1}) \bmod q$

Ali-DSA Variant [23]	$s = (k(x.H(M)+r))^{-1}) \bmod q$

According to the difference in calculating the value of signature *s*, the verification equations for each variant are also different in number of computation steps and involved arithmetic operations. After calculating the hash value of the received message *H(M)*, the verifier uses the corresponding equations for each DSA-variant as shown in table 3, to calculate the value of *v*, which serves as the signature validation parameter.

Table 3: Verification Equations for DSA Variants

Algorithm	Verification Equations
GOST-DSA variant[20]	$v = H(M)^{q-2} \bmod q$ $z_1 = (s.v) \bmod q$ $z_2 = ((q-r).v) \bmod q$ $u = ((g^{z_1}.y^{z_2}) \bmod p) \bmod q$
Yen-Laih-DSA variant [21]	$w = r^{-1} \bmod q$ $u_1 = (w.H(M)) \bmod q$ $u_2(w.s) \bmod q$ $v = ((g^{u_1}.y^{u_2}) \bmod p) \bmod q$
McCurley-DSA variant [22]	$u_1 = (H(M).s) \bmod q$ $u_{21} = (s.r) \bmod q$ $v = ((g^{u_1}.y^{u_2}) \bmod p) \bmod q$
Ali-DSA variant [23]	$u_1 = (H(M)+r) \bmod q$ $u_2 = (s.u_1) \bmod q$ $v = (y^{u_2} \bmod p) \bmod q$

Therefore, for these verification equations if the obtained value for *v* is equal to *r*, then the sender is accepted as authentic and the integrity of the message is verified, otherwise it is rejected.

4. THE PROPOSED MODIFIED DIGITAL SIGNATURE ALGORITHM (M-DSA)

To increase the speed of DSA algorithm it is better to reduce the most time consuming processes such as modular inverse calculations, modular exponentiation, and modular multiplication as much as possible, but without affecting the algorithm strength. This describes the proposed modified version of DSA algorithm (named M-DSA). It involves fewer modular operations at the signature verification process than those used in NIST-DSA, with the aim of improving the overall execution time of the algorithm, while keeping the security at the same level of the standard NIST-DSA. The proposed message signing and its signature validation processes are outlined below, together with their differences from the standard DSA.

4.1 **Signing Process**

As stated earlier, the signing of any message *M* consists of the calculation two signature parameters, r and s, hence after calculating the hash function *H(M)*, these parameters are calculated by equation 9.

$$r = (g^k \bmod p) \bmod q \qquad (9\text{-a})$$

$$s = (r + k(x.H(M))^{-1}) \bmod q \qquad (9\text{-b})$$

Comparing equations (9-a and 9-b) with those listed in section 2 for the NIST-DSA and its variants, it can be seen that the value of *r* is exactly the same as that used for all DSA variants, but the involved modification here is obviously shown in the calculation of *s*. Although equation 9-b contains the same parameters *r*, k, x, and *H(M)* that are used in other DSA variants, but their operations are modified and differs from all other variants. This modification in is require for the correctness proof for the signature validation explained in the lemma.

Now the signature *r* and *s,* together with the message *M* can be send over to the receiver using any communication media.

4.2 Signature Validation Process

At the receiver side the intended recipient will validate the signature of the sender for the received message in order to accept or reject its authenticity. The recipient calculates the hash value of the received message *M* then uses the sender public key (*y*) and other public parameters (i.e. *p*, *q*, and *g*) which he/she already knows in order to validate the message signature using the following equations 10-12.

$$u_1 = s.H(M) \bmod q \qquad (10)$$
$$u_2 = r.H(M) \bmod q \qquad (11)$$
$$v = (y^{(u_1 - u_2)} \bmod p) \bmod q \qquad (12)$$

These equations are arranged differently from the validation equations of NIST-DSA and all other DSA variants, see table 3. They are modified in this arrangement in order to fulfil the requirement for the signature validation, as will be clarified by the mathematical proof of proposed M-DSA algorithm, shown in the lemma of section 4.3.

Now if the calculated *v* is equal to the received *r*, then the message integrity is verified and the signature is validated.

Investigation of M-DSA signature validation equations (i.e. equations 10-12), shows that they are less complex and only contains one exponentiation in *v* value calculation, which results into faster verification process than NIST-DSA and other variants (eg. GOST, Yen-Laih, and McCurley).

4.3 Lemma

Given a value of *r* as in equation 1 and calculating *v* using equation 12, then for the message signature verification to be true, the calculated value of *v* must equal to *r*. Therefore in order to get *v*=*r*, one must prove that the exponents in equations 1 and equation 12 must be equal, i.e.

$$k = x.u_1 - x.u_2 \qquad (13)$$

Proof

The message receiver, or the signature verifier calculates first calculates *H(M)*, u_1, and u_2 as follows:

$u_1 = s.H(M) \bmod q$, and $u_2 = r.H(M) \bmod q$

But *v* is calculated by equation 12.

i.e.

$$v = (y^{u_1 - u_2} \bmod p) \bmod q$$

And $\quad y = (g^x \bmod p) \bmod q$

Now substitution *y* into *v* gives:

$$v = (g^{x.(u_1 - u_2)} \bmod p) \bmod q$$
$$= ((g^{x.u_1 - x.u_2}) \bmod p) \bmod q \qquad (14)$$

But equation 1 is

$$r = (g^k \bmod p) \bmod q \qquad (1)$$

Therefore, if the calculated value for *v* equals to the required *r* (i.e. *v* = *r*), then one gets

$$(g^k) \bmod p) \bmod q = ((g^{x.u_1 - x.u_2}) \bmod p) \bmod q \qquad (15)$$

For equation 15 to be valid, it must be proved that the exponents of *g* on both sides must be equal.

i.e. $\quad k = x.(u_1 - u_2) \qquad (16)$

Substitution of u_1 and u_2 into the right side of equation 16 results into equation 17.

$$x.(u_1 - u_2) = (x(s.H(M) - r.H(M)) \bmod p) \bmod q$$
$$= (x.H(M)(s - r) \bmod p) \bmod q \quad ..(17)$$

Then substitution of *s* from equation 9 into equation 17, the right side becomes

$$= (x..H(M)(r + k.H(M))^{-1} - r) \bmod p) \bmod q$$
$$= (k \bmod p) \bmod q = k$$

Hence, $\quad x.(u_1 - u_2) = k \qquad$ (q.e.d)

5. RESULTS AND DISCUSSION

5.1 Time Complexity

It is natural that one of the most required factors for message authentication and message integrity is to get the message signature and/or validate that signature or doing both operations in the shortest possible period of time.

As the time complexity is the main concern in this research work, therefore Big-O values for both signing and signature verification processes of the proposed M-DSA, together with those for NIST-DSA and the other variants described earlier are determined and compared with each other as listed in table 4.

Table 4: Time Complexity Comparison

No.	Algorithm	BigO		
		Signature	Verification	Total
1	NIST-DSA	2 log n	3 log n	5 log n
2	GOST-DSA	log n	3 log n	4 log n
3	McCurley-DSA	2 log n	2 log n	4 log n
4	Yen-Laih-DSA	2 log n	3 log n	5 log n
5	Ali-DSA	2 log n	log n	3 log n
6	Proposed M-DSA	2 log n	log n	3 log n

It can be deduced from Table 4, that the proposed algorithm and that of Ali-DSA variants have the lowest Big O values, which indicate that the proposed M-DSA will have improved time complexity. To experimentally support this achievement, the proposed modified digital signature algorithm, together with all considered algorithms are coded, implemented, and tested under the same computing environment (namely using the same personal computer with CPU clock = 2.7 GHz vPro Intel® processor core i7 and programming in C# language and Python software).

The time complexities for the signing and signature verification processes by the proposed M-DSA algorithm are computed and recorded using various parameters lengths and message contents, then compared with those computed for all other considered algorithms. The obtained time complexity results are plotted in Figure 2. These results represent the average time for measurements which is taken over 900 signatures and 900 verifications over 15 rounds.

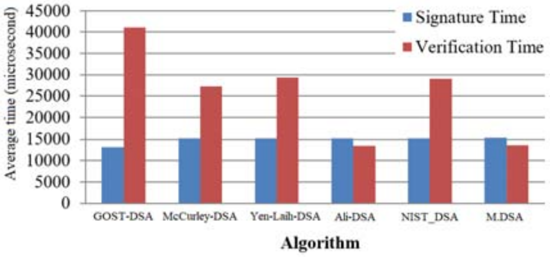

Fig 2: Comparison of Total Average of Signing and Verification Time

Table 4 and figure 2 illustrate that the signature verification speed for the proposed algorithm M-DSA, together with Ali-DSA have the fastest signature verification speed than other algorithm except Ali-DSA which has almost the same speed. On the other hand, GOST algorithm has the fastest signing speed of all algorithms considered but the slowest verification speed. However, the signing time for M-DSA algorithms is similar to those of DSA and the others except GOST-DSA which has the lowest signing time. Therefore, the proposed algorithm looks more suitable in situations where verification time is crucial factor, while the signing time might be tolerated.

These results suggest that an efficient digital signature algorithm variant that has speed improvement in both signing and verification might be possible if the GOST-DSA is coupled with the proposed M-DSA. However, this idea needs further research efforts and will be followed by the authors.

5.2 Test M-DSA with NIST Primes p and q

Using data values of the prime integers p, q, and h, provided by the Cryptographic Algorithm Validation Program Testing for Digital Signatures (CAVP) web site [24], M-DSA is tested and the time complexity measurements for the total of signing and verification processes are obtained. The used values of these primes were with length same as described in NIST-DSA are listed in table A1 of the appendix, for length of p and q equal to 1024 bits and 160 bits, respectively. This test is repeated 900 times for each signing and verification processes for both M-DSA and NIST-DSA, then the average is found and listed in table 5.

Table 5: Average Processing Time for Signing and Verification Processes

Algorithm	Average processing time (µ seconds)	
	Signing	verification
NIST-DSA (Standard)	303806.8078	95877.39778
M-DSA (Proposed)	305011.1322	48006.08667

5.3 Security Concerns

The proposed M-DSA is a digital signature algorithm based on NIST-DSA, it computes r and s values as the signature using the hash value of the message H(M) rather than the message M itself, and verifies the signature using public key of the sender as reported in the proof of the lemma in section 4.3. Hence, the message is sent as plaintext with its signature, providing message integrity and sender authentication only. This is exactly equivalent to the hand written signature, the digital signature algorithm does not encrypt the signed message, but uses a mechanism to authenticate the sender and to ensure that the message was not altered during transmission. At the signing side, beside the sender's secret key, a random integer is used with each message, then verification is done to this signature using only the sender's public key, hence the sender cannot deny his signature on the message, which provide non-repudiation. Furthermore, if message secrecy is required, any cryptographic technique can be implemented for encryption and decryption of the message besides the digital signature algorithm.

5.4 Key Strength

The digital signature algorithms strength depends on the length of the prime integers, the length of the private key, and the length of the randomly generated secret integer. The length in bits for standard DSA for p is between 1024 to 3027 bits, and for q is 160 to 256 bits. Nowadays the speed of the computers allows using larger values of the primes p and q which are recommended to provide higher security against the attackers. It is noticed that key strength for the proposed M-DSA is the same as that for the standard NIST_DSA due to the fact that it employs the same set of parameters, and relies on modular arithmetic processes.

6. SPEED GAINS

The proposed M-DSA has provided an efficient time saving during the signature verification process as compared with DSA algorithm and its other variants which is considered as a good improvement. However, signing process time did not have any changes. A speed gain factor is defined below in order to illustrate the time complexity improvement in the verification process. For example, the speed gain (G) in the signature verification achieved by M-DSA over NIST-DSA is obtained by equation 18.

$$G = \frac{VT_{DSA}}{VT_{M-DSA}} \qquad (18)$$

Where, VT_{DSA} and $VT_{M\text{-}DSA}$ are the verification time for DSA and M-DSA algorithm, respectively.

Likewise, the speed gain for the proposed M-DSA with respect to the other DSA variant algorithms under consideration are evaluated for over 1000 run, then the average speed gain for the signature verification is listed in Table 6.

Table 6: Average Verification Speed Gain for M-DSA over other Algorithm variants.

Signature verification time		VT_{M-DSA} (µsec)	Average G
Algorithm	VT (µsec)		
NIST-DSA	29092.6	13644.08	2.13
GOST-DSA	41062.09	13644.08	3.00
McCurley-DSA	27255.33	13644.08	1.99
Yen-Laih-DSA	29286.99	13644.08	2.14
Ali-DSA	13493.13	13644.08	0.98

Table 5 shows that the modified digital signature algorithm M-DSA, has manifested a considerable speed gain over other DSA algorithm variants under consideration. This improvement resulted from the reduction in the number of exponentiation operations in the verification process. However, it must be admitted that no speed gain was obtained on the signature process, therefore this algorithm would be useful for applications where, signature verification time is crucial.

7. CONCLUSIONS

In conclusion, the modified DSA algorithm (M-DSA) has shorten the message signature validation by about half the time needed for the standard DSA, as shown in Table 5. This improvement is attributed to the reduction in the number of modular arithmetic processes. Therefore, the signature verification speed for the M-DSA is almost double those for NIST-DSA, Yen-Laih, and McCurley, and three times that for GOST-DSA, but equals that for Ali-DSA. However, no change is notice in the signing speed of the standard DSA algorithm and its variants less GOST which was the fastest of all. Besides, this enhancement in the processing speed was achieved with the same key strength of the standard DSA due to the used of the same parameters.

Therefore, it can be concluded that the proposed M-DSA is more suitable for applications where verification time is crucial factor either due to the limited power resources in remote stations or battlefield combat applications. M-DSA has the same properties and same security level of DSA since that both algorithms depend on hardness of discrete logarithm problem (DLP), where their security depends on the length of the used primes numbers selected as the public keys.

ACKNOWLEDGMENT

The authors are grateful to the Middle East University, Amman, Jordan for the financial support granted to cover the publication fee of this research article.

REFERENCES

[1] Menezes, A. J., Van Oorschot, P. C., & Vanstone, S. A. (1996). Handbook of applied cryptography. CRC press.

[2] Fu, C., & Zhu, Z. L. (2008). An efficient implementation of RSA digital signature algorithm. In 2008 4th International Conference on Wireless Communications, Networking and Mobile Computing (pp. 1-4). IEEE.

[3] Chang, X. (2009). PDFeH: A PDF Based Generic Teacher-Student E-Homework System. In Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on (pp. 1-4). IEEE.

[4] Schneier, B., (1996). Applied cryptography: protocols, algorithms, and source code in C. New York: Wiley.

[5] Tan, C. H., Yi, X., & Siew, C. K., 2003. Signature scheme based on composite discrete logarithm. In Information, Communications and Signal Processing, 2003 and Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint Conference of the Fourth International Conference on (Vol. 3, pp. 1702-1706). IEEE.

[6] Poulakis, D. (2009). A variant of digital signature algorithm. Designs, codes and cryptography, 51(1), 99-104.

[7] Hernandes, P. R. G., & Carvalho, L. F. (2014, November). Digital Signature of Network Segment Using Flow Analysis through Genetic Algorithm and ACO Metaheuristics. In Chilean Computer Science Society (SCCC), 2014 33rd International Conference of the (pp. 92-97). IEEE.

[8] Singh, M., Kaur, H., & Kakkar, A. (2015). Digital signature verification scheme for image authentication. In 2015 2nd International Conference on Recent Advances in Engineering & Computational Sciences (RAECS) (pp. 1-5). IEEE.

[9] Tan, Y., Tang, S., Chen, J., Yu, Y., & Li, X. (2016). Building a new secure variant of Rainbow signature scheme. IET Information Security, 10(2), 53-59.

[10] Andrade, E. R., & Terada, R. (2016). Proposal of Enhancement for Quartz Digital Signature. Brazilian Journal of Information Security and Cryptography, 2(1), 3-15.

[11] Dhagat, R., & Joshi, P. (2016). New approach of user authentication using digital signature. In Colossal Data Analysis and Networking (CDAN), Symposium on (pp. 1-3). IEEE.

[12] Al Imem, A., (2015). Comparison and Evaluation of Digital Signature Schemes Employed In NDN Network. International Journal of Embedded systems and Applications (IJESA) Vol. 5, No. 2, June.

[13] Manickam, S., & Kesavaraja, D. (2016). Secure multi server authentication system using elliptic curve digital signature. In Circuit, Power and Computing Technologies (ICCPCT), 2016 International Conference on (pp. 1-4). IEEE.

[14] Stallings, W. (2013). Cryptography and Network Security: Principles and Practice.

[15] Engström, Pontus (2016). "Modernizing forms at KTH: Using Digital Signatures.", M.Sc. degree thesis In Information and Communication Technology, First Cycle, Stockholm, Sweden.

[16] Fu, C., & Zhu, Z. L. (2008). An efficient implementation of RSA digital signature algorithm. In 2008 4th International Conference on Wireless Communications, Networking and Mobile Computing (pp. 1-4). IEEE.

[17] Diffie, W., & Hellman, M. (1976). New directions in cryptography. IEEE transactions on Information Theory, 22(6), 644-654.

[18] ElGamal, T. (1984). A public key cryptosystem and a signature scheme based on discrete logarithms. In Workshop on the Theory and Application of Cryptographic Techniques (pp. 10-18). Springer Berlin Heidelberg.

[19] NIST.FIPS.186-4, http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf.

[20] Popov, V., Kurepkin, I., and S. Leontiev, (2006), "Additional Cryptographic Algorithms for Use with GOST 28147-89, GOST R 34.10-94, GOST R 34.10-2001, and GOST R 34.11-94 Algorithms", RFC 4357, January.

[21] Laih C. S., and S. M. Yen, (1995), “Improved Digital Signature Algorithm”, IEEE Transactions on Computers, Vol. 44, No. 5, PP 729-730.

[22] Kevin S. McCurley, “Digital Signature Standard”,

[23] http://www.mccurley.org/papers/health/node13.html, last visited Aug 2017.

[24] Ali, H. A. (2004). Improved Verification Speed Enhancement for Digital Signature Using Discrete Algorithm Variant. Journal of the Association of the Advancement of Modeling and Simulation Techniques in Enterprises (AMSE).

APPENDIX

The values for the parameters implemented for the standard NIST-DSA and provided by the Cryptographic Algorithm Validation Program Testing for Digital Signatures (CAVP) web site [24] are listed in table A-1. The same parameter values are also used for the developed M-DSA tests for comparison purposes.

Table A-1. Test Parameters for NIST-DSA and M-DSA

parameter	Value used

p	1097053912923206937871140030510676164 68
	259212779421298214601125674928181254497
	258412808557904798486074233461755793872
	404301860507251869207922097495749515369
	440166311897766141731380397656565036353
	495745303138237011627541307378142311900
	036257047321025499964025765613745066612
	99339575651467003124931404843562936 9

q	1222224633501956391786986258814797381563 3625555641
h	4175243646082476646142906352247815353 39 320230946
x	Random integer with length 48 decimal digit
k	Random integer with length 306 decimal digit