TEST Engineering & Management

May – June 2020
ISSN: 0193-4120 Page No. 2254 - 2260

A New Method for Constructing Digital Signature Algorithm based on a New Key Scheme

[1] Van Luu Xuan, [2] Binh Nguyen Luong, [3] Dung Luu Hong
[1] People's Security Academy, [2] Military Technology Academy, [3] Military Technology Academy
[1] vanlx.hvan@gmail.com, [2] nluongbinh@yahoo.co.uk, [3] luuhongdung@gmail.com

Article Info
Volume 83
Page Number: 2254 - 2260
Publication Issue:
May - June 2020

Article History
***Article Received**: 11August 2019*
***Revised**: 18November 2019*
***Accepted**: 23January 2020*
***Publication**: 10 May2020*

Abstract:

E-governance is the latest trend in many countries in which the government is using the online system for delivering government services to citizensand the Digital Signature is used to validate contents and authorize users who are involving in the E-governance system. Although there are many digital signature schemes currently known as: RSA, Elgamal,... the approach of improving the safety level of digital signature schemes based on the hardness of solving simultaneous difficult assumptions has still remained underdeveloped and attracted rising attentions from researchers. In this research, with the target to improve safety level of digital signature algorithms, we present the proposed method for constructing digital signature algorithms which is based on a new key scheme. This new key format is constructed using a new difficulty problem without high-efficiency solution and it can be applied to construct digital signature algorithms. According to this research, a high-security digital signature class can be construct by the proposed method in some actual situation.

Keywords: *Digital Signature Algorithm, Public Key Cryptosystem, Discrete Logarithm Problem, Elgamal Cryptosystem.*

I. INTRODUCTION

Nowadays, in the development of Industry 4.0, a lot of information exchange and control activities have been carried out over the network. Along with the convenience, information security is potentially risky, increasingly important and needs special attention. All online transactions need to ensure the integrity and identity of the sender of the message. Therefore, digital signatures will be widely used in the near future. In addition to applying classic digital signature algorithms such as RSA, Elgamal [12], ..., scientists have researched and developed other digital signature algorithms to meet in some specific cases, for example improving safety or improving the efficiency of the algorithm. As research improves the security of digital signature algorithms, a number of studies have investigated the development of difficult problems in number theory. By combining the solvability of these difficult problems, some new digital signature algorithms have been constructed and developed.

In [1-11], several studies have combined hard problems of discrete logarithm and integer factorization to develop digital signature algorithms. In line with this research direction, we propose a method to build a new digital signature algorithm based on a new key scheme. The new key scheme was developed from a root problem and discrete logarithm problem, which is difficult to find an effective solution. Therefore, digital signature scheme based on this new key scheme is secure, and can be applied in many actual situations.

II. CONSTRUCTING DIGITAL SIGNATURE ALGORITHM BASED ON A NEW KEY SCHEME

A. *Proposed key scheme*

Key scheme of Elgamal digital signature algorithms are built based on the discrete logarithm problem:

$$y = g^x \bmod p \tag{1}$$

Here, (x, y) are private key and public key of a sign, (g, p) are domain parameters, system parameters that were created by digital authentication service provider, in which, g is generator of the group Z_p, p is

May – June 2020
ISSN: 0193-4120 Page No. 2254 - 2260

a prime number. Finding private key x, from public parameters y, g and p, is a difficult problem if p is a large and strong prime number.

From (1), it can be seen that if g is also a secret parameter then (1) will becomes the form:

$$y = (x_1)^{x_2} \bmod p \quad (2)$$

Finding exactly one pair of values (x_1, x_2) from public parameters (y, p) will be a difficult problem that has not been solved yet (except "brute fore attack"). Theoretically, if (2) is used as the key generation scheme for the digital signature algorithm, in which, (x_1, x_2) are private keys and y is the signer's public key, the security of key will be higher than key of Elgamal signature algorithm.

However, using (2) to create keys also has some certain weaknesses, for example, an attacker who use the pair (y, 1) can still create a valid digital signature without having to find the exact value of the private keys (x_1, x_2). The reason is that, according to (2), the values of x_1 and x_2 can be chosen independently of each other, so, the attacker can choose $x_2 = 1$ and easily obtain $x_1 = y$.

To overcome such weaknesses of (2), we propose a new key scheme as following: With (p_1, p_2) are prime numbers that being the domain parameters/system parameters that created by the sender of service provider, and they must satisfy the condition $p_2 | (p_1 - 1)$, each object chooses a number $\alpha \in (1, p_1)$. Let sk be private key, it is generated by equation (3):

$$sk = \alpha^{(p_1-1)/p_2} \bmod p_1 \quad (3)$$

Then, the public key is generated, according to (4), as follows:

$$pk = (sk)^{(sk)^{-1} \bmod p_2} \bmod p_1 \quad (4)$$

The key scheme here can be described in algorithm 1 as a key generation algorithm as follows:

Algorithm 1. Key Generation Algorithm

Input: l_1, l_2 - length by bits of p_1, p_2.

Output: p_1, p_2, sk, pk.

1: **Generate** p_1, p_2: *length*(p_1) = l_1, *length*(p_2) = l_2, $p_2 | (p_1 - 1)$

2: **Select** α : $\alpha \in (1, p)$

3: $sk \leftarrow \alpha^{(p_1-1)/p_2} \bmod p_1$

4: **If** ($sk = 1$) **then goto** 2

5: $pk \leftarrow (sk)^{(sk)^{-1} \bmod q} \bmod p$

6: **Return** $\{p_1, p_2, sk, pk\}$

Note:

- *length*(.): function calculates the length by bits of an integer number.
- p_1, p_2 : prime numbers (system parameters) are selected.
- *sk*: private key of signature object (S).
- *pk*: public key of signature object (S).

When using (3) and (4) as a key generation scheme for the digital signature algorithm, we must construct right signing and testing algorithms. The following section will present a method of constructing digital signature algorithm based on a new key scheme.

B. Constructing digital signature algorithm based on a new key scheme

A digital signature scheme consists of three main algorithms: Key Generation Algorithm, Signing Algorithm and Signature Verifying Algorithm. The method of constructing digital signature scheme will be presented immediately following:

The Key generation algorithm is described in Algorithm 1, the construction of Signing algorithm and Signature verifying algorithm will be presented in the following:

1. Signing algorithm

Suppose (S_1, S_2) is the signature of message M and S_1 is calculated from $s \in Z_p^*$ by the following formula:

$$S_1 = (s)^{(sk)^{-1}} \bmod p_1 \quad (5)$$

and S_2 is calculated from $t \in Z_p^*$ by:

$$S_2 = (t)^{(sk)^{-1}} \bmod p_1 \quad (6)$$

Assume that the signature verifying equation of the algorithm has the form:

$$S_1^{\,H} \equiv S_2^{\,pk} \times pk^{(S_1 \times S_2 \bmod p_1) \bmod p_2} \bmod p_1 \quad (7)$$

In which:

- $H = Hash(M)$, *Hash*(.) is a hash function;
- $(S_1 \times S_2 \bmod p_1) \bmod p_2 = \left((b)^{(sk)^{-1}} \bmod p_1\right) \bmod p_2$;
- $b = \beta^{(p_1-1)/p_2} \bmod p_1$, $\beta \in (1, p_1)$.

Let:

$$\left((b)^{(sk)^{-1}} \bmod p_1\right) \bmod p_2 = Z. \quad (8)$$

So, the signature verifying equation will be changed into the following form:

$$(S_2)^H \equiv (S_1)^{pk} \times (pk)^Z \bmod p_1. \quad (9)$$

From (4), (5), (6) and (9), we have:

$$(t)^{(sk)^{-1}.H} \equiv (s)^{(sk)^{-1}.pk} \times (sk)^{(sk)^{-1}.Z} \bmod p_1. \quad (10)$$

From (10) to:

May – June 2020
ISSN: 0193-4120 Page No. 2254 - 2260

$$t = (s)^{pk.(H)^{-1}} \times (sk)^{Z.(H)^{-1}} \bmod p_1 . \quad (11)$$

Otherwise, from (5), (6) and (7), we have:

$$s \times t \bmod p_1 = b . \quad (12)$$

From (11) and (12), we have:

$$s \times (s)^{pk.(H)^{-1}} \times (sk)^{Z.(H)^{-1}} \bmod p_1 = b .$$

or:

$$(s)^{(H)^{-1}.pk+1} \times (sk)^{Z.(H)^{-1}} \bmod p_1 = b . \quad (13)$$

From (10), we have:

$$s = \left(b \times (sk)^{-Z.(H)^{-1}}\right)^{\left((H)^{-1}.pk+1\right)^{-1}} \bmod p_1 . \quad (14)$$

From (14) and (12), we can calculate t by:

$$t = b \times (s)^{-1} \bmod p_1 .$$

From the values s and t, we can calculate the value of S_1, a component of the signature, by equation (5):

$$S_1 = (s)^{(sk)^{-1} \bmod p_2} \bmod p_1 .$$

and calculate S_2, the second component of the signature, by equation (6):

$$S_2 = (t)^{(sk)^{-1} \bmod p_2} \bmod p_1 .$$

Here, signing algorithm can be described in Algorithm 2 as follows:

Algorithm 2. Signing Algorithm

Input: p_1, p_2, sk, pk, M.
Output: (S_1, S_2).

1: $H \leftarrow Hash(M)$

2: **Select** β : $\beta \in (1, p_1)$, $b \leftarrow \beta^{\frac{p_1-1}{p_2}} \bmod p_1$

3: $Z \leftarrow \left((b)^{(sk)^{-1} \bmod p_2} \bmod p_1\right) \bmod p_2$

4: $s \leftarrow \left(b \times (sk)^{-Z.(H)^{-1} \bmod p_2}\right)^{\left((H)^{-1}.pk+1\right)^{-1} \bmod p_2} \bmod p_1$

5: $t \leftarrow b \times (s)^{-1} \bmod p_1$

6: $S_1 \leftarrow (s)^{(sk)^{-1} \bmod p_2} \bmod p_1$

7: $S_2 \leftarrow (t)^{(sk)^{-1} \bmod p_2} \bmod p_1$

8: **If** ((S_1=1) **or** (S_2=1)) **then goto** 2

9: **Return** (S_1, S_2)

Note:

- M: Message need to be signed, $M \in \{0,1\}^{\infty}$.
- *Hash*(.): Hash function $Hash: \{0,1\}^{*} \mapsto Z_n$, in which: $p_2 < n < p_1$.
- (S_1, S_2): Signature of S on M.

2. Signature verifying algorithm

Verifying formula of this proposed algorithm is assumed that:

$$(S_2)^H \equiv (S_1)^{pk} \times (pk)^{(S_1 \times S_2 \bmod p_1) \bmod p_2} \bmod p_1 .$$

In which, M is message that must be verified and H is a hash value of the message M ($H=Hash(M)$). The recipient will authenticate the integrity and originality of the message M based on its signature. If the message M is consistent with its signature (S_1, S_2), the recipient can trust the message. In other words, the above expression must be true. Otherwise, the message or the signature is invalid and its correctness is not be recognized. So, we can calculate the left side of signature verifying formula as:

$$V_1 = (S_2)^H \bmod p_1 . \quad (15)$$

and we can calculate the right side of signature verifying formula as:

$$V_2 = (S_1)^{pk} \times (pk)^{\bar{Z}} \bmod p_1 . \quad (16)$$

in which:

$$\bar{Z} = (S_1 \times S_2 \bmod p_1) \bmod p_2 . \quad (17)$$

So, if $V_1=V_2$, then the signature is valid and the message will be verified by originality also integrity.

Therefore, the signature verifying algorithm of the proposed scheme is described in Algorithm 3 as:

Algorithm 3. Signature Verifying Algorithm.

Input: p_1, pk, M, S_1, S_2.
Output: *True / False*.

1: $H \leftarrow Hash(M)$

2: $V_1 \leftarrow (S_2)^H \bmod p_1$

3: $\bar{Z} \leftarrow (S_1 \times S_2 \bmod p_1) \bmod q_2$

4: $V_2 \leftarrow (S_1)^{pk} \times (pk)^{\bar{Z}} \bmod p_1$

5: **If** $(V_1 = V_2)$ **then** {**return** *True*}
else {**return** *False*}

3. The correctness of the proposed scheme

Assessing the validity of the proposed scheme can be stated as follows:

Let (p_1,p_2) are two prime numbers with $(p_1 - 1) \vdots p_2$, a hash function $Hash: \{0,1\}^{*} \mapsto Z_n$, $p_2 < n < p_1$, $\alpha \in (1, p_1)$,

$sk = \alpha^{(p_1-1)/p_2} \bmod p_1$, $pk = (sk)^{(sk)^{-1} \bmod p_2} \bmod p_1$,

$\beta \in (1, p_1)$, $b = \beta^{(p_1-1)/p_2} \bmod p_1$,

$Z = \left(b^{(sk)^{-1} \bmod p_2} \bmod p_1\right) \bmod p_2$, $H = Hash(M)$,

$s = \left(b \times (sk)^{-Z.(H)^{-1} \bmod p_2}\right)^{\left((H)^{-1}.pk+1\right)^{-1} \bmod p_2} \bmod p_1$,

$t = (s)^{pk.(H)^{-1} \bmod p_2} \times (sk)^{Z.(H)^{-1} \bmod p_2} \bmod p_1$,

$S_1 = (s)^{(sk)^{-1} \bmod p_2} \bmod p_1$, $S_2 = (t)^{(sk)^{-1} \bmod p_2} \bmod p_1$.

If $\bar{Z} = (S_1 \times S_2 \bmod p_1) \bmod p_2$, $V_1 = (S_2)^H \bmod p_1$, $V_2 = (S_1)^{pk} \times (pk)^{\bar{Z}} \bmod p_1$ then: $V_1 = V_2$.

TEST Engineering & Management

May – June 2020
ISSN: 0193-4120 Page No. 2254 - 2260

Assessing the validity of the proposed algorithm can be proved as follows:

From equation (4), (5), (6), (11), (14) and (16), we have:

$$\begin{aligned} V_1 &= (S_2)^H \bmod p_1 = (t)^{(sk)^{-1}.H} \bmod p_1 \\ &= \left((s)^{pk.(H)^{-1}} \times (sk)^{Z.(H)^{-1}}\right)^{(sk)^{-1}.H} \bmod p_1 \\ &= (s)^{(sk)^{-1}.pk} \times (sk)^{(sk)^{-1}.Z} \bmod p_1 \\ &= (S_1)^{pk} \times (pk)^Z \bmod p_1. \end{aligned} \tag{18}$$

Moreover, from equation (5), (6), (11), (14) and (15), we have:

$$\begin{aligned} \bar{Z} &= (S_1 \times S_2 \bmod p_1) \bmod p_2 \\ &= \left((s)^{(sk)^{-1}} \times (t)^{(sk)^{-1}} \bmod p_1\right) \bmod p_2 \\ &= \left((s)^{(sk)^{-1}} \times \left(b \times (s)^{-1}\right)^{(sk)^{-1}} \bmod p_1\right) \bmod p_2 \\ &= \left((b)^{(sk)^{-1}} \times (s)^{(sk)^{-1}} \times (sk)^{-(sk)^{-1}} \bmod p_1\right) \bmod p_2 \\ &= \left((b)^{(sk)^{-1}} \bmod p_1\right) \bmod p_2 = Z. \end{aligned} \tag{19}$$

From (19) and (16), we have:

$$V_2 = (S_1)^{pk} \times (pk)^{\bar{Z}} \bmod p_1 = (S_1)^{pk} \times (pk)^Z \bmod p_1. \tag{20}$$

From (18) and (20), we have the proof: $V_1 = V_2$. Thus, the above algorithm is correct.

C. Example

The correctness of the algorithm that is constructed by the new proposed method has been proved above, the following example continues to be a confirmation of the correctness of this algorithm.

1. Parameter selection and Key generation (Algorithm 1):

- A prime number p_1:
7696527888022625042701519518043291787188858223210604292619149826755089643880961400280237187617444292577200732246948641191946554666178703515956898081270671

- A prime number p_2:
137318612304891229417393653643373002193471933 9773

- Value of sk:
2300658867134420904756563401425244398155990818121705379973725934174799666467163626001650127291002057553657175582837273663124479073641172724995112153441509

- Value of pk:
2841990085655257167368269424609515142867746651288729520154718851405600234080898233431704583310521767820285451033738234439317209690947612929624884251717220

2. Signature generation (Signing algorithm - Algorithm 2):

Input: p_1, p_2, sk, pk, M.

Output: (S_1, S_2).

- Message M:
"THIS IS A NEW METHOD FOR CONSTRUCTING DIGITAL SIGNATURE ALGORITHMS BASED ON A NEW KEY SCHEME!"

- Value of b:
306554396852891752065566546500117236257563592030360251928672988983163690105996812382440620226246315110104898510423132997130914045131171674153738478390881 6

- Value of H:
95936638572933842689397875108618980925643180 7060

- Value of S_1:
222244178677216728286052898648623725677370843895082356890066313659814030966535708518080199703161131181731590581990044984635886214810253038806305730734728

- Value of S_2:
254543455868811978663422223643198117378606977217206660290253453708613022868560005632096432067870564031253044814884137263687415546423303970028047368884619 0

3. Signature Verification (Signature verifying algorithm - Algorithm 3):

Input: $p_1, p_2, pk, (S_1, S_2), M$.

Output: *True/ False*.

+ Case No.1:

- The message M:
"THIS IS A NEW METHOD FOR CONSTRUCTING DIGITAL SIGNATURE ALGORITHMS BASED ON A NEW KEY SCHEME!"

- Value of S_1:
222244178677216728286052898648623725677370843895082356890066313659814030966535708518080199703161131181731590581990044984635886214810253038806305730734728

- Value of S_2:
254543455868811978663422223643198117378606977217206660290253453708613022868560005632096432067870564031253044814884137263687415546423303970028047368884619 0

- Value of H:
95936638572933842689397875108618980925643180 7060

- Value of Z:
311117531159940372943481648083994702272462903994188256441003511245862817556543609369642413957442155639843155177694699061827456192686864231693123182414044 5

- Value of V_1:
196153530260495640166720063614764116008322276938558751842389057820267102587920589495523684620777363307724604946477376620538168346667471369427777418579044 5

- Value of V_2:
196153530260495640166720063614764116008322276938558751842389057820267102587920589495523684620777363307724604946477376620538168346667471369427777418579044 5

Output: (S_1, S_2) = *True*.

In this case, all message and signature were not modified during transmission, so, with the above result, the message and its signature are consistent, and so that, the recipient can trust the integrity and original of the message.

+ Case No.2: The message M has been modified by an attacker.

- The message M (fake):
"THIS IS A OLD METHOD FOR CONSTRUCTING DIGITAL SIGNATURE ALGORITHMS BASED ON A NEW KEY SCHEME!"

- Value of S_1:
222244178677216728286052898648623725677370843895082356890066313659814030966535708518080199703161131181731590581990

TEST Engineering & Management

May – June 2020
ISSN: 0193-4120 Page No. 2254 - 2260

044984635886214810253038806305730734728

- Value of S_2:
2545434558688119786634222236431981173786069772172066602902534537086130228685600056320964320678705640312530448148841372636874155464233039700280473688846190

- Value of H:
49789245265502077531030000076484782224926234320

- Value of Z:
3111175311599403729434816480839947022724629039941882564410035112458628175565436093696424139574421556398431551776946990618274561926868642316931231824140445

- Value of V_1:
315292313694979225659730572221399007614560559754199368955372764691045957675543283167562733312975854117709072269909784004189635353390700047656178816002322

- Value of V_2:
1961535302604956401667200636147641160083222769385587518423890578202671025879205894955236846207773633077246049464773766205381683466674713694277774185790445

Output: $(V_1, V_2) = False$.

In this case, the message (M) has been modified (the attacker replaces the word "NEW" with the word "OLD"), with the above result, the message and its signature are not consistent, and so that, the recipient should not trust the integrity and original of the message. Be careful!

+ Case No.3: S_1 of digital signature is not correct.

- The message M:
"THIS IS A NEW METHOD FOR CONSTRUCTING DIGITAL SIGNATURE ALGORITHMS BASED ON A NEW KEY SCHEME!"

- Value of S_1:
222244178677216728286052898648623725677370843895082356890066313659814030966535708518080199703161131181731590581990044984635886214810253038806305730734720

- Value of S_2:
2545434558688119786634222236431981173786069772172066602902534537086130228685600056320964320678705640312530448148841372636874155464233039700280473688846190

- Value of H:
959366385729338426893978751086189809256431807060

- Value of Z:
5837282506162320564465597143513972994002645532197162619047208296034855277723519843969421136997109311629790163327061933099120982211540435262558136557182938

- Value of V_1:
1961535302604956401667200636147641160083222769385587518423890578202671025879205894955236846207773633077246049464773766205381683466674713694277774185790445

- Value of V_2:
979542255861402849556167937885455592577650818417500972704378472361292124832848476123404673668799481946932006186111343478079418307938198873649409093046173

Output: $(S_1, S_2) = False$.

In this case, a component of the signature (S_1) has been modified (some errors may occur during sending process), according to the above result, the message and its signature are not consistent, and so that, the recipient should not trust the integrity and original of the message. Be careful!

+ Case No.4: S_2 of digital signature is not correct.

- The message M:
"THIS IS A NEW METHOD FOR CONSTRUCTING DIGITAL SIGNATURE ALGORITHMS BASED ON A NEW KEY SCHEME!"

- Value of S_1:
222244178677216728286052898648623725677370843895082356890066313659814030966535708518080199703161131181731590581990044984635886214810253038806305730734728

- Value of S_2:
2545434558688119786634222236431981173786069772172066602902534537086130228685600056320964320678705640312530448148841372636874155464233039700280473688846199

- Value of H:
959366385729338426893978751086189809256431807060

- Value of Z:
5111372919694354284009292568677560553820966634997623776420631935396954454264257470359145936902871737034015867014857395479997537860160919666187983400752997

- Value of V_1:
6621073992715407829464174846835252183617991275537335034104075283021029562808622750559432277396223830945453533535627133034288013314905653061978850859602873

- Value of V_2:
5259160678546272186594733722077862201253347461959528801016018068297040447629968411323462191131784575568525183583673101212954715647992906348413748801089422

Output: $(S_1, S_2) = False$.

In above case, a component of signature (S_2) has been modified (the last digit of S_2 has been changed), according to the above result, the message and its signature are not consistent, and so that, the recipient should not trust the integrity and original of the message.

III. SECURITY LEVEL OF THE PROPOSED AGORITHM BASED ON THE NEW KEY SCHEME

The safety level of the proposed algorithm based on the new key scheme can be assessed by its ability to resist some types of attacks such as:

- *Safety against attacks on the private key*

From (4), finding *sk* by algorithms for Discrete Logarithm Problem on Z_p such as Pollard Rho, Baby Step - Giant Step, Index Calculus, ... is impossible. Nowaday, the "brute force attack" is the only algorithm that can be applied to solve (4). However, "brute force attack" is not a polynomial time algorithm that can be applied in practical applications.

- *Safety against attack on signing algorithm*

An attacker could perform an attack on the signing

May – June 2020
ISSN: 0193-4120 Page No. 2254 - 2260

algorithm to find *sk*. Finding *sk* from solving equations:

$$S_1 = (s)^{(sk)^{-1} \bmod p_2} \bmod p_1 .$$

or:

$$S_2 = (t)^{(sk)^{-1} \bmod p_2} \bmod p_1 .$$

or:

$$Z = \left((b)^{(sk)^{-1} \bmod p_2} \bmod p_1\right) \bmod p_2 .$$

(In which: $Z = (S_1 \times S_2 \bmod p_1) \bmod p_2$)

Including to solve equation (4), this is a hard problem. It shows that attacking signing algorithm to find *sk* is a completely impossible.

+ Safety against attack on signature verifying algorithm

From the Signature verifying Algorithm (Algorithm 3) of the proposed algorithm, if attacker want to forge a signature, he must create a fake signature (S_1, S_2) will be recognized as a valid signature of a message M, in other word, it must satisfies the following conditions:

$$(S_2)^E \equiv (S_1)^{pk} \times (pk)^{(S_1 \times S_2 \bmod p_1) \bmod p_2} \bmod p_1 . \qquad (21)$$

From (21), if S_1 is preselected and then S_2 will be calculated, the condition (21) will become the following form:

$$(S_2)^H \equiv (X)^{S_2} \bmod p_1 . \qquad (22)$$

If S_2 is preselected and then S_1 will be calculated, the condition (21) will become the following form:

$$(S_1)^{pk} \equiv (Y)^{S_1} \bmod p_1 . \qquad (23)$$

With X and Y being constants, it is easy to see that (22) and (23) are hard problems that currently do not have a solution. It is easy to see that if the left side of (22) and (23) is a constant number then this is a Discrete Logarithm Problem, and if the right side is a constant number then this is the Root Problem. However, in (22) and (23), both sides contain variables to be searched, so, algorithms for Discrete Logarithm Problem and Root Problem on Z_p are not applicable to solve (22) and (23).

IV. CONCLUSION

The paper proposes a new method to build a digital signature algorithm based on a new key scheme. By using this new key scheme, we can develop some digital signature algorithms. These algorithms are high secure and can be applied in some actual situation.

REFERENCES

[1] Wei-Hua He, “Digital signature scheme based on factoring and discrete logarithms”, in Electronics Letters, volume 37, no. 4, pp. 220-222, 15 Feb 2001. DOI: 10.1049/el:20010149
[2] Shiang-Feng Tzeng, Cheng-Ying Yang and Min-Shiang Hwang, “A new digital signature scheme based on factoring and discrete logarithms”, in International Journal of Computer Mathematics, volume 28, issue 1, pp. 9-14, 2004. DOI: 10.1080/00207160310001614954.
[3] Li-Hua Li, Shiang-Feng Tzeng and Min-Shiang Hwang, “Improvement of signature scheme based on factoring and discrete logarithms”, in Applied Mathematics and Computation, volume 161, pp. 49-54, 2005. DOI: 10.1016/j.amc.2003.12.008.
[4] Shimin Wei, “Digital Signature Scheme Based on Two Hard Problems”, in International Journal of Computer Science and Network Security, volume 7, no. 12, pp. 207-209, 2007.
[5] Nikolay A. Moldovyan, “Digital Signature Scheme Based on a New Hard Problem”, in Computer Science Journal of Moldova, volume 16, no. 2(47), pp. 163-182, 2008.
[6] Eddie Shahrie Ismail, Nedal Tahat and Rokiah Rozita Ahmad, “A New Digital Signature Scheme Based on Factoring and Discrete Logarithms”, in Journal of Mathematics and Statistics, volume 4, issue 4, pp. 222-225, 2008. DOI: 10.3844/jmssp.2008.222.225.
[7] Al-Fayoumi Mustafa, Aboud Sattar and Al-Fayoumi Mohammad, “A New Digital Signature Scheme Based on Integer Factoring and Discrete Logarithm Problem”, in International Journal Computer Applications, volume 17, no. 2, pp. 108-115, 2010.
[8] Nedal Tahat, Zead Mustafa and A. K. Alomari, “New ID-Based Digital Signature Scheme on Factoring and Discrete Logarithms”, in Applied Mathematical Sciences, volume 6, no. 28, 1363-1369, 2012.
[9] Andrey N. Berezin, Nikolay A. Moldovyan and Victor A. Shcherbacov, “Cryptoschemes Based on Difficulty of Simultaneous Solving Two Different Difficult Problems”, in The Computer Science Journal of Moldova, volume 21, no. 2 (62), 2013.
[10] Dimitrios Poulakis, “A public key encryption scheme based on factoring and discrete logarithm”, in Journal

of Discrete Mathematical Sciences and Cryptography, volume 12, no. 6, pp. 745-752, 2013. DOI: 10.1080/09720529.2009.10698270.

[11] Shin-Yan Chiou, "Novel Digital Signature Schemes based on Factoring and Discrete Logarithms", in International Journal of Security and Its Applications, volume 10, no. 3, pp. 295-310, 2016. DOI: 10.14257/ijsia.2016.10.3.26.

[12] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms", in IEEE Transactions on Information Theory, volume 31, no. 4, pp. 469-472, 1985. DOI: 10.1109/TIT.1985.1057074.