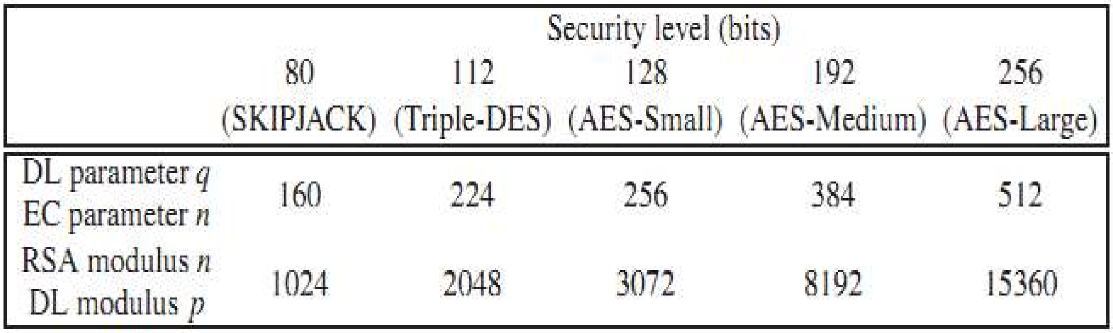
According to the RSA signature scheme with small exponent, in publication [6] in 1979, M.O Rabin proposed a key signature scheme with an exponent e = 2. The choice of e = 2 has brought the outstanding advantage that Rabin's signature-verifying algorithm only requires a modulo squared operation. In the RSA scheme, with $n = p \times q$ $(p, q$ are primes), we have $gcd\big(e, \phi(n)\big) = 1$ $(with\ \phi(n) = (p-1) \times (q-1))$. In the Rabin scheme, by choosing e=2, we clearly have e as the division of $\phi(n)$, which means the parameters p, q satisfy $p \equiv q \equiv 1\ (mod\ 2)$. When proposing a signature scheme [6], Rabin noted that if there is $p \equiv q \equiv 1\ (mod\ 3)$, it is possible to replace the congruent quadratic equation with the congruent cubic equation to produce a signature scheme with safety ensured by the factorizing problem.



	Security level (bits)				
	80 (SKIPJACK)	112 (Triple-DES)	128 (AES-Small)	192 (AES-Medium)	256 (AES-Large)
DL parameter q EC parameter n	160	224	256	384	512
RSA modulus n DL modulus p	1024	2048	3072	8192	15360

Figure 1. *Security level*

In 1980, Williams improved the version of the Rabin scheme, abbriaviated to RW [7]. This scheme only required one Jacobi symbol calculation while the Rabin scheme required 4 Jacobi symbol calculations for creating the signatures. With such ultimate feature, it was brought into ISO/IEC 9796 standard in 2002 [8]. In 1986, based on the improvements made on the Rabin scheme in the RW scheme [7], Williams proposed a RW-based signature scheme with an exponent e = 3.

Apart from the research of Williams [7] [9] [10], there were numerous other research on improving the RSA scheme with small exponent, such as the authors J. H. Loxton, David S. P. Khoo, Gregory J. Bird and Jennifer Seberry [11], Scheidler [12],... These results constructed a version of the RSA scheme with e=3 and larger class of primes p and q.

In [13], we proposed a deterministic signature scheme combining RSA and Rabin for the case where p - *1* is a multiple of 3 and q - *1* is primate with 3. The proposed scheme cost little verification time as a 3 exponent modulo.

On solving the problem of constructing a signature scheme with low signature-verifying cost for digital transaction that require authentication of signature validity in a great deal, in this paper, the we propose a probabilistic signature scheme based on RSA and Rabin with the exponent e = 3. The organization of this paper is organized as follows. In section II, the paper presents the mathematical basis of the signature scheme as the open problem the square root on Z_p with p is the prime number larger than 3. In section III and IV, a probabilistic signature scheme and the correctness and safety of the proposed scheme are presented. Finally, section V summarizes obtained results and future research.

2. CONTENT

2.1. Mathametic base

2.1.1. Symbols

With all a $\in \mathbb{Z}_{\mathrm{n}}$ corresponding only with $(\mathrm{a_p}, \mathrm{a_q}) \in \mathbb{Z}_{\mathrm{p}} \times \mathbb{Z}_{\mathrm{q}}$ with $\mathrm{a_p} = a \bmod p$ and $\mathrm{a_q} = a\ mod\ q$ and reverse mapping, denoted as *CRT*, is determined by the formula:

$$CRT(u,v) = (q.(q^{-1} \bmod p).u + p.(p^{-1} \bmod p).v) \bmod n \tag{1}$$

- Mapping on the preservation of multiplication means:

$$CRT(u.x \bmod p, v.y \bmod q) = CRT(u,v).\ CRT(x,y) \bmod n \tag{2}$$

2.1.2. Some additive results.

a) **Lemma 1:** With the prime $\boldsymbol{p = t.k^s + 1}$ with gcd(t,k) = 1, denoted as :

$$u = -\mathrm{t}^{-1} \bmod k; \tag{3}$$

Then d defined by

$$d = \frac{\mathrm{u(p-1) + k^s}}{\mathrm{k^{s+1}}}$$

is an integer.

Proof

As $p = t.k^s + 1$, we have:

$$d = \frac{u(p-1) + k^s}{k^{s+1}} = \frac{u.t.k^s + k^s}{k^{s+1}} = \frac{u.t + 1}{k}$$

According to (3):

$$ut = -\mathrm{t}.\mathrm{t}^{-1}\ mod\ k = -1 \bmod k$$

then

$$ut = xk - 1 \text{ with random integer x.}$$

Then, we have:

$$d = \frac{xk - 1 + 1}{k} = x$$

So, d is an integer.

b) The value p and d when k=3

In the case of s=1, then p = t.3 + 1 with gcd(t,3) = 1

We have t ≡ 1, 2 (mod 3).

More specifically, we have:

- With t ≡ 1 (mod 3) then p ≡ 4 (mod 9). According to (3), we have u ≡ 2 (mod 3)
- With t ≡ 2 (mod 3) then p ≡ 7 (mod 9). According to (3), we have u ≡ 1 (mod 3)

Then d is obtained by using the help of (4).

$$d = \begin{cases} \frac{2p+1}{9} \; with\, p \equiv 4 \text{ (mod 9)} \\ \frac{p+2}{9} with\, p \equiv 7 \text{ (mod 9)} \end{cases}$$

c) Definition 1 (Function *CR,* where the letters CR stand for "Cube Root")

Given p ≠ 1 (mod 9) as an odd prime, we have:

$$d = \begin{cases} 3^{-1} \bmod (p - 1) & \text{nếu } p \neq 1 \text{ (mod 3)} \\ \frac{2p+1}{9} & \text{nếu } p \equiv 4 \text{ (mod 9)} \\ \frac{p+2}{9} & \text{nếu } p \equiv 7 \text{ (mod 9)} \end{cases} \quad (5)$$

Function CR (., p): GF(p) → GF(p) is determined by the following formula:

$$CR(a,p) = a^d \bmod p. \quad (6)$$

with GF(p), where the letters GF stand for "Galois field", is a finite field that is given by the integers mod p when p is a prime number.

Then, we have :

Lemma 2. With p ≠ 1 (mod 9) as an odd prime, then with a ∈ GF*(p) we have :

If *p ≠ 1 (*mod *3)* then

$$CR(a,p)^3 \equiv a \text{ (mod } p). \quad (7)$$

If *p ≡ 4 (*mod *9)* then

$$CR(a,p)^3 \equiv a.\left(a^{\frac{p-1}{3}}\right)^2 \text{(mod } p). \quad (8)$$

If *p ≡ 7 (*mod *9)* then

$$CR(a,p)^3 \equiv a.a^{\frac{p-1}{3}} \text{ (mod } p). \quad (9)$$

Lemma 3.

Considering the equation below with a ∈ $\mathbb{Z}_n$.

$$x^3 \equiv a \ (\text{mod } n). \tag{10}$$

We have results as follows.

Conditions needed and sufficient for (10) to have a solution:

$$a^{\frac{p-1}{3}} \bmod p = 1 \tag{11}$$

Then, a solution of (10) is given by the following formula:

$$x = \text{CRT}\ (\text{CR}\ (a \bmod p, p), \text{CR}\ (a \bmod q, q)). \tag{12}$$

Corollary 1. If n can be analyzed into factors p and q, then equation (10) always be solved.

To the best of authors knowledge, there has not been any publication that indicates a solution of (10) can be found without knowing the analysis of n. In contrast, there has not been a claim that n can be analyzed if the equation (8) can be solved. Here, this paper would give a possibly closest result that can solve the opposite problem as follows.

Clause 1. If two different solutions of equation (10) are found, then n can be analyzed.

Time cost for arthmetic operation on Z_n.

The cost of running time some algorithms performing arithmetic operations:

(1) The cost of adding or subtracting two k-bits is O (k) [14, pp. 30-31].

(2) Multiplying two 2.k-bits integers by the method of Karatsuba-Ofman requires three k-bit double multiplications [14, p. 51]. At this time the multiplication cost, denoted by M, and the cost for squaring two large numbers, denoted by S, are approximately equal (M ≈ S). Moreover, we also get the cost for multiplication:

$$M = O(k^{ln3/ln2}).$$

Formula 13. *M (k) is the computation cost to perform a multiplication of two k-bit integers. Then with all positive integers h, k we have:*

$$\text{M(h)} \approx 3^{\text{t}}.\text{M(k), with t} = \log_2\left(\frac{\text{h}}{\text{k}}\right). \tag{13}$$

A 2.k-bit truncation algorithm in one modulo k-bit using Barrett's math requires two multiplication of k-bit numbers [14, p. 36]. Infer the value $\text{t} = \log_2\left(\frac{\text{h}}{\text{k}}\right) = 1.$. Therefore, the result is:

Formula 14: *The cost of implementing a reduced multiplication in modulo n is approximately* (14)

(3) Jacobi symbol-calculating algorithm for a number modulo k-bit base on the rule of reciprocal square has complexity as $O(k^2)$ [15, p. 98].

(4) The calculation cost of the inverse of a number in modulo k-bit, denoted by I, and the cost of dividing a number by modulo k-bits, denoted by D, according to Sorenson's algorithm has complexity O (k ^ 2 / lnk) [15, pp. 463-465].

(5) According to formula (1), performing the CRT function requires two inverse modulo p and a division modulo n. According to the results shown in Table 2.1, the cost of the inverse modulo k - bit is equal to the cost of a division modulo k - bit and that cost is k^2/lnk. So the time cost for a CRT function calculation is $3.k^2/lnk$.

The time cost for arithmetic operations on Zn is summarized in the following table.

Table 1. The cost of runtime of arithmetic operations on Z_n

Operation	Complexity	Algorithm
The cost of adding or subtracting two k-bit integers	O(k)	[14, pp. 30-31]
The cost for multiplying two k-bit integers	$M(k) = O\left(k^{ln3/ln2}\right)$	[14, p. 51].
The cost for ashortened multiplication in modulo n	3.M(len(n))	[14, p. 36]
Calculating Jacobi symbol of a number in modulo k-bit according to the law of reciprocal squares	$O(k^2)$	[15, p. 98].
The calculation cost of the inverse of number modulo k-bits	$O(k^2/lnk)$	[15, pp. 463-465].
The cost for divide a number by modulo k-bits	$O(k^2/lnk)$	[15, pp. 463-465].

2.2. Signature scheme PCRS

As stated in the introduction, this paper proposes signature schemes that have a low cost for verifying algorithms for use in one-stop transactions. The Rabin and RSA schemes, with exponent e as small as possible, have the feature above. In this section, the paper proposes a probabilistic signature scheme, namely PCRS. Similar to the Rabin scheme, the parameters p, q of PCRS satisfy the condition $p \equiv q \equiv 1\ (mod\ 3)$.

2.2.1. Systematic parameter

System parameter for signature schemes includes:

– Integer n = p.q with p, q are two primes so that:

  – *p = 3.t + 1* with *gcd(t,3) = 1*
  – *and q = 3.k + 1* with *gcd(k,3) = 1* (15)

– *Hash* Function: $\{0,1\}^\infty \rightarrow \{0,1\}^h$ satisfies security requirements for codes.

– Secret parameter d_p, d_q can be defined as follows:

$$d_p = \begin{cases} \frac{2p+1}{9} \; if\ p \equiv 4 \pmod 9 \\ \frac{p+2}{9} \; if\ p \equiv 7 \pmod 9 \end{cases};$$

(16)

$$- \quad d_q = \begin{cases} \frac{2q+1}{9} \; if\ q \equiv 4 \pmod 9 \\ \frac{q+2}{9} \; if\ q \equiv 7 \pmod 9 \end{cases};$$

2.2.2. Signing message.

Algorithm 1

Input: M ∈ $\{0,1\}^{\infty}$ is the message to be signed.

Output: (R,s) ∈ $\{0,1\}^{k} \times \mathbb{Z}_n$ is the signature in M.

```
1.  Repeat
    R = Random({0,1}^k);
    h = Hash(R||M);                                                  (17)
            $t = h^{\frac{p-1}{3}} \bmod p; u = h^{\frac{q-1}{3}}\ mod\ q;$
    until (t==1) and (u==1)
2.  $h_p = h \bmod p;\ h_q = h \bmod q;$                              (18)
4.  $s_p = h_p^{d_p} \bmod p;\ s_q = h_q^{d_q} \bmod q;$              (19)
5.  $s = CRT(s_p, s_q);$                                              (20)
6.  return (R, s);                                                    (21)
```

2.2.3. Verifying Signature.

Algorithm 2

Input: M ∈ $\{0,1\}^{\infty}$; (R,s) ∈ $\{0,1\}^{k} \times \mathbb{Z}_n$ is the signature in M.

Output: *Accept* ∈{0,1} only accept the validity of the signature if and only if *Accept* = 1.

```
1.      h = Hash(R||M);
2.      t = $s^3$ mod n;
3.      Accept = (t==h);
        return Accept;
```

2.2.4. The correctness of the signature scheme

The correctness of the PCRS scheme is given by the following result:

Clause 2. *All signatures (R, s) on text M created from algorithm 1 have an output value of 1 according to algorithm 2.*

Proof:

According to step 1 of algorithm 1 that creates a signature, we have t = 1 and u = 1 so from the formula (17) we have:

$$- \quad h^{\frac{p-1}{3}} \bmod p = 1 \qquad (22)$$

Since h satisfies (22), the equation: $s^3 \equiv h \pmod n$ always has solution $s = CRT(s_p, s_q)$

According to formula (19) we have:

$$s_q^3 \bmod q = (h_q^{d_q})^3 \bmod q = (h_q^{3d_q}) \bmod q$$

And :

$$s_p^3 \ mod\ p = (h_p^{d_p})^3 mod\ p = (h_p^{3d_p}) mod\ p$$

– If p ≡ 4 (mod 9) then $3d_p = \frac{2p+1}{3} = 1 + 2\frac{p-1}{3}$, so:

$$s_p^3\ mod\ p = (h_p^{3d_p}) mod\ p = h_p^{\left(1+2\frac{p-1}{3}\right)} mod\ p$$
$$= h_p . h_p^{2\frac{p-1}{3}} mod\ p = h_p . \left(h_p^{\frac{p-1}{3}}\right)^2 \bmod p = h_p$$

– If p ≡ 7 (mod 9) then $3d_p = \frac{p+2}{3} = 1 + \frac{p-1}{3}$, so:

$$s_p^3 \bmod p = (h_p^{3d_p}) \bmod p = h_p^{\left(1+\frac{p-1}{3}\right)} \bmod p$$
$$= h_p . h_p^{\frac{p-1}{3}} \bmod p = h_p$$

Similarly, we have: $s_q^3 \bmod q = h_q$

So:

$$\begin{cases} s_p^3 \bmod p = h_p \\ s_q^3 \bmod q = h_q \end{cases}$$

From formula (20) we have:

$$\begin{aligned} s^3 = \left(CRT(s_p, s_q)\right)^3 &= CRT(s_p^3\ mod\ p, s_q^3 \bmod q) \\ &= CRT(h_p, h_q) = h \end{aligned} \qquad (23)$$

From the result gained from (23) we have a comparison (t == h) in step 3 of the PCSR1 signature-verifying algorithm that always returns a result of 1 (true). Therefore, *Accept =1.* This is what needs to be proven.

2.3. Time cost to run the PCRS schemes

2.3.1. Calculation cost of PCRS scheme

The cost of the signing algorithm is calculated based on the steps to conduct the algorithm and the size of the input parameters. Therefore, with size of modulo p guaranteeing security given in figure 1, we consider the size of parameters d_p, d_q based on formula (19)

According to formula (19), we have:

– With $p \equiv 4 \pmod 9$ then $d_p = \frac{2\mathrm{p}+1}{9} < \frac{p}{4}$. Thus, the size of d_p is smaller than the size of modulo p at least 2 bits, which means $len(d_p) \leq \mathrm{len}(p) - 2$.

– With $p \equiv 7 \pmod 9$ then $d_p = \frac{\mathrm{p}+2}{9} < \frac{p}{8}$. Thus, the size of d_p is smaller than the size of modulo p at least 3 bits, which means $\mathrm{len}(d_p) \leq \mathrm{len}(p) - 3$

– We have equivalent result with parameter d_q.

– So, the size of d_p and d_q is 2 to 3 bits smaller than the size of p, q. To simplify the calculation while guarantee the validity of the calculation of time cost for running algorithm, we can choose the case of maximum length of d_p and d_q , which is equal to $\mathrm{len}(p)$.

Considering the calculation cost of the signature-creating algorithm (algorithm 1)

– In step 1, the algorithm executes the loop with the stop condition t = 1 and u = 1 with the power of h being the cube root of the unit. Since the probability of finding a third-degree surplus is $\frac{1}{3}$, so in loop 1 of algorithm 1 we need to do it in the $3^2 = 9$ times of the exponentiation. We denote t_{exp} as the time cost for a power calculation, then the time cost for step 1 is approximately $9.t_{exp}$

– In step 2, two modulo are operated so the cost is $2.\ln(\mathrm{len}(p))$.

– In step 3, two exponentiations are operated so the cost is $2.t_{exp}$

– In step 4, a CRT function calculation is needed, we denote it as t_{CRT}.

Considering the computational cost of the signature-verifying algorithm (algorithm 2)

– In step 2, a power of 3 on $\mathbb{Z}_\mathrm{n}$ is required (by 2 multiplications). We denote t_m as the time cost of performing a multiplication, then the time cost of step 2 is $2.t_m$.

From the above analysis, we obtain the time cost of the PCRS scheme as follows:

The cost of the signature creation algorithm, denoted as T_1, and the test algorithm, denoted as T_2, in the PCRS sheme are given by the following formula:

$$T_1 = 11 \times t_{exp} + 2.\ln\big(len(p)\big) + t_{CRT} \tag{24}$$

$$T_2 = 2 \times t_m \tag{25}$$

According to the square-multiplication method, the average time cost for u exponentiation is calculated by multiplication $t_{exp} = 1.5\, len(u).t_m$.

Moreover, according to the results given in table 1, the cost of multiplying two integers of which lengths is k - bit is $k^{ln3/ln}$. Therefore $t_m = k^{ln\ /ln2}$.

According to formula (1), performing the CRT function requires two inverses of modulo p and a division by modulo n. According to the results shown in table 1, the cost of the inverse in modulo k - bit is equal to the cost of a division by modulo k - bit and that cost is k^2/lnk. So, the time cost for a CRT function calculation is $3.k^2/lnk$.

With the security-guaranteed size modulo p given in figure 1 and the parameter d_p given by the formula (19), we have the same size parameter d_p of size modulo p.

Clause 4. *The cost of the signature-creation algorithm, denoted as T_1,, and the verifying algorithm, denoted as T_2, in the PCRS scheme are given by the following formula:*

$$T_1 = 16.5 \times \mathrm{len}(p) \times \mathrm{len}(p)^{\frac{ln3}{ln2}} + 2.\ln\big(\mathrm{len}(p)\big) + \frac{3\mathrm{len}(p)}{\ln \mathrm{len}(p)}$$

Then:

$$T_1 = 16.5 \times \mathrm{len(p)}^{(\frac{\ln 3}{\ln 2}+1)} + 2.\ln\big(\mathrm{len}(p)\big) + \frac{3\mathrm{len}(p)}{\ln \mathrm{len}(p)} \tag{26}$$

And:

$$T_2 = 2 \times \mathrm{len}(p)^{\frac{ln3}{ln2}} \tag{27}$$

2.3.2. The effectiveness of the proposed schemes

The PCRS scheme is probabilistic, combined with the principles of the RSA and Rabin schemes in case of e=3.

In loop 1 of algorithm 1, we need to perform averagely $3^2 = 9$ times (because the probability to find a third-degree surplus is $\frac{1}{3}$). In general, with a similar development of Rabin for verifying exponent that is prime e, the complexity of the signature-creating algorithm will require averagely e^2 times in the loop to find an e-degree surplus. Thus, the PCRS scheme's signature-creating algorithm has greater complexity than Rabin's because the Rabin scheme needs to perform averagely four times in the loop.

2.4. Security of the schemes PCRS

To create a valid signature onto document M depending on the scheme PCRS, a counterfeiter has two choices:

Firstly, from the public parameter n, the counterfeiter has to find two prime numbers p, q with n=p.q. With p and q known, dp and dq are found, therefore counterfeit signature can be created. To do this, the counterfeiter has to solve the factorizing problem.

Secondly, if there is no secret parameter (p,q), the counterfeiter has to find a root of the cubic congruent equation depending on modulo n. The finding of a root of equation (10) is called "CRP-Cube Root Problem".

According to Consequence 1, if n if analyzed into factors p and q, then the cubic congruent equation can always be solved depending on modulo n. And according to Clause 1, if two different roots of the equation (10) are found, then n can be analyzed.

So, the security of the signature schemes is guaranteed by the difficulty of the factorizing problem and CRP problem. The factorizing problem was proven by digital theory to be difficult and is the mathematic base for a lot of the encryption systems. The CRP problem was proven by Consequence 1 to be led from the factorizing problem. This is why the signature scheme PCRS is safe.

3. CONCLUSION

In this paper, we have suggested a signature scheme based on the probabilistic model that is developed on the RSA and the Rabin scheme in case of e=3, in which the PCRS scheme is similar to the Rabin but with e=3. The PCRS is a combination between RSA and Rabin when e=3 is the divisor of p-1 but coprime with q-1. The significant features of the suggested schemes in this paper are algorithm creating signatures without calculating Jacobi symbol and signature-testing algorithm based on the Rabin scheme (which leads to its low cost). These diagrams have security based on the difficulty of analyzing n in the sense that if n can be analyzed, the schemes can be broken. So far, there is no effective algorithm (polynomial time) that breaks one of the signature diagrams mentioned above.

The signature schemes suggested in this paper guarantee safety and have low verifying cost. This is one of the vital features so that the digital signature schemes can be implemented in digital administrative and commercial services.

REFERENCES

1. FIP 186-2 (2000), Digital Signature Standard (DSS). Federal Information Processing Standards Publication 186-2. National Institute of Standards and Technolog.

2. NIST 800-56, NIST Special Publication 800-56: Recommendation on key establishment schemes, Draft 2.0.

3. Darrel Hankerson, Alfred Menezes, and Scott Vanstone (2004), *Guide to El-liptic Curve Cryptography*, Springer-Verlag.

4. A. LENSTRA (2001), "Unbelievable security—matching AES security using public key systems.," *Advances in Cryptology— A SIACRYPT 2001 (LNCS 2248),* vol. 67, pp. 67-86.

5. Arjen K. Lenstra and Eric R. Verheul (2001), "Selecting Cryptographic Key Sizes," *Journal of Cryptology.,* vol. 14, pp. 255-293.

6. M. O. Rabin (1979), "Digitalized signatures and Public-Key Functions as Intractable as Factorization," *Foundations of Secure Computations, Academic Press New York.*

7. H. C. Williams (1980), "A modification of the RSA public-key encryption procedure”," *IEEE Transactions on Information Theory ISSN 0018–9448,* vol. 26, pp. 726-729.

8. ISO/IEC 9796, "International Standard ISO/IEC 9796 Information technology – Security techniques- Information Security risk management".

9. ISO/IEC 9796, International Standard ISO/IEC 9796 Information technology – Security techniques- Information Security risk management.

10. H. C. Williams (1985), "Some public-key crypto-function as intractable as factorization," *Cryptologia,* vol. 9, no. 3, pp. 223-237.

11. H. Loxton, David S. P. Khoo, Gregory J. Bird and Jennifer Seberry (1992), "A Cubic RSA Code Equivalent to Factorization," *Journal of Cryptology,* vol. 5, pp. 139-150.

12. R. Scheidler (1998), "A Public-Key Cryptosystem Using Purely Cubic Fields," *Journal of Cryptology,* vol. 11, p. 109–124.

13. H. T. Mai (2018), "The Signature Scheme In Combination With RSA And RABIN," *Journal of Military Science and Technology,* vol. 53, pp. 143-148.

14. Darrel Hankerson, Alfred Menezes, and Scott Vanstone (2004), Guide to El-liptic Curve Cryptography, Springer-Verlag.

15. Richard Crandall, Carl Pomerance (2005), Prime Number A Computational Perspective, Springer.

PHÁT TRIỂN CHỮ KÝ SỐ RSA VÀ RABIN VỚI SỐ MŨ E=

Tóm tắt: *Sơ đồ chữ ký số RSA và sơ đồ chữ ký số Rabin đều là các lược đồ chữ ký được xây dựng trên cơ sở tính khó giải của bài toán phân tích số. Nếu như số mũ xác thực trong chữ ký số RSA là e phải thỏa mãn $gcd(e,\phi(n)) = 1$ thì trong hệ Rabin $e=2$ và luôn là ước của $\phi(n)$. Theo hướng kết hợp giữa RSA và Rabin, bài báo đề xuất lược đồ chữ ký theo mô hình xác suất cho trường hợp số mũ xác thực $e= 3$ và 3 là ước của $\phi(n)$.*

***Từ khóa*:** *RSA Signature Scheme, Digital Signature Scheme, Rabin Signature Scheme, Cube Root Signature Scheme.*

AN INTUITIVE SOFTWARE FOR TEACHING AND LEARNING CYBER ATTACKS

Tong Anh Tuan

People's Police University of Technology and Logistics

Abstract: *Currently, cybersecurity issues are of great concern, and at the same time, this major is being increasingly trained at universities in the country. In this article, we outline some common types of cyber-attacks such as SQL Injection, Cross Site-Scripting, Denial of Service, and introduce a software that simulates those attacks through objects, processes, and illustrations. This software supports teachers as well as students in the process of teaching and learning.*

Keywords: *Teaching tool, SQL Injection, Cross Site-Scripting, Denial of Service.*

Received: 26 March 2020

Accepted for publication: 21 April 2020

Email: tuanqb92@gmail.com

1. INTRODUCTION

At present, the information security field is increasingly interested in training at universities. This area covers the security of network and information infrastructure, computer security, data, and application software, as well as research and development of public products and technological solutions. According to the project, "Project of training and developing human resources for information security until 2020" sets out the task of training human resources at 08 key training institutions on information security and producing more than 2,000 high-quality information security personnel who graduate from universities or higher education institutions [1].

About information security teaching, students need to understand cyber-attack techniques in general and each stage and specific principles of each technique in particular. Typically, some common cyberattacks nowadays include SQL Injection (SQLi), Cross Site-Scripting (XSS), Denial of Service (DOS) [2]. To show the principles and stages of these techniques, the conventional method used is to present the theory, interpret steps through diagrams, drawings, or practice specific Lab exercises.

In this article, we introduce the Cyber Demonstration software, teaching, and learning support tool. Accordingly, the software will simulate and illustrate the principles and steps