

# The Variant of Digital Signature Algorithm for Constant Message

Kritsanapong Somsuk<sup>1\*</sup>, Sarutte Atsawaraungsuk<sup>2</sup>, Chanwit Suwannapong<sup>3</sup>,  
Suchart Khummanee<sup>4</sup> and Chalida Sanemueang<sup>5</sup>

<sup>1</sup>Department of Computer and Communication Engineering, Faculty of Technology, Udon Thani Rajabhat University, UDRU, Udon Thani, Thailand.  
kritsanapong@udru.ac.th, Orcid: <https://orcid.org/0000-0002-7264-4628>

<sup>2</sup>Department of Computer Education, Udon Thani Rajabhat University, Udon Thani, Thailand. sarutte@udru.ac.th,  
Orcid: <https://orcid.org/0000-0002-6853-6480>

<sup>3</sup>Department of Computer Engineering, Faculty of Engineering, Nakhon Phanom University, Thailand. schanwit@npu.ac.th,  
Orcid: <https://orcid.org/0000-0002-7970-3088>

<sup>4</sup>Department of Computer Science, Faculty of Informatics, Mahasarakham University, Thailand. suchart.k@msu.ac.th, Orcid: <https://orcid.org/0000-0002-6078-1203>

<sup>5</sup>Office of Academic Resources and Information Technology, Udon Thani Rajabhat University, UDRU, Udon Thani, Thailand. Chali.sa@udru.ac.th,  
Orcid: <https://orcid.org/0009-0001-2223-6909>

Received: February 18, 2023; Accepted: March 30, 2023; Published: May 30, 2023

## Abstract

This study aims to present a modified technique for signing constant messages. In general, intruders may often steal the digital signature of a constant message with relative ease. Assuming there is a constant message that must always be signed by the signer, the digital signature must equally have a constant value. If it is communicated through an insecure channel to the recipient or verifier and is intercepted along the way by attackers, they can assume the identity of the signer and use this signature for authentication. In fact, the proposed method, Digital Signature Algorithm for Constant Message (DSACM) and DSACMV2, are the result of the combination between RSA and OTP. In addition, OTP is selected for signing and validating procedures in which the secret key must be regenerated for each process. Thus, the ciphertext is constantly changing, but the message remains fixed. Moreover, RSA is chosen to protect the transmission of the secret key across an insecure channel. The experimental findings indicate that DSACM and DSACMV2 are suitable for signing a message with a constant value because the signature is an undetermined value. Although it takes two encryption procedures and two decryption processes, the time required to generate the secret key and perform the exclusive or operation increases little. In addition, the proposed methods have the benefit that the constant message is not modified. In fact, it must be combined with an integer such as a timestamp and a random number for the other techniques for changing the ciphertext, and it cannot be signed a single time if its length exceeds the private key.

---

*Journal of Internet Services and Information Security (JISIS)*, volume: 13, number: 2 (May), pp. 81-95.  
DOI: [10.58346/JISIS.2023.12.005](https://doi.org/10.58346/JISIS.2023.12.005)

\*Corresponding author: Department of Computer and Communication Engineering, Faculty of Technology, Udon Thani Rajabhat University, UDRU, Udon Thani, Thailand.

**Keywords:** DSACM, RSA, OTP, Digital Signature.

## 1 Introduction

The process of encrypting and decrypting data is known as cryptography (Farah, R.S., 2020), and it is used to protect data sent through an in secured channel. The structure is split in half. The first part is called symmetric key cryptography. The secret key is required on both of sender and receiver sides. The advantage of symmetric key cryptography is the reduced processing time. In general, many efficient algorithms, such as AES (Zhang, Y., 2018) and (Mathe, S.E., 2017) and One Time Pad (OTP) (Shihua, Z.A., 2021) and (Ali, S., 2021), require a little amount of time to complete the task. However, the problem of the technique is the exchange of the secret key between the sender and the recipient. In fact, the second part, known as asymmetric key cryptography or public key cryptography (Diffie, W., 1976) was developed to address this problem. For the second portion, two relatively mathematically different keys are required: the public key and the private key. While the public key can be shared publicly, the private key must be kept secretly. Choosing the algorithm for exchanging the secret key was a standard practice in the earliest days of public key cryptography. However, it is currently also a feasible alternative for data encryption and digital signature (Koblitz, N., 1987), (Miller, V.S., 1986) and (ElGamal, T., 1985). In fact, the method for verification is a digital signature (Jafar, A.A., 2021). To authenticate the digital message, the signer will employ the private key to sign the message. On the other hand, the signed message is verified by using the public key. RSA (Rivest, R.L., 1978) is the algorithm of public key cryptography for both data encryption and digital signature (Farah, J.A., 2018). In general, the digital signature of the constant message is used for many tasks. From the authentication, for instance, the ability to operate the device through the internet of thing (IoT) is selected. Assume ‘on’ and ‘off’ are the constant messages to turn on and turn off the light, respectively. Prior to placing an order for the device, the signer must sign one of the constant messages to obtain the digital signature before transmitting to the device (verifier). The device will then verify the digital signature before carrying out its specified operation. However, if RSA is chosen to implement the digital signature and the message remains constant, it is quite simple for intruders to steal this digital signature along the route. Therefore, they can then use this value for authentication. In fact, adding a timestamp or a random value to a constant message to generate a distinct signature is an example of a technique for solving this problem. However, if the message is larger than the private key, the signature process may be repeated. Furthermore, if the message that is a combination of a constant message and an integer is chosen to command the device via IoT, after the verification process is complete, algorithms are required to extract only the constant message.

The aim of the research is to propose a variant of the Digital Signature Algorithm for Constant Message (DSACM) for signing constant messages. In truth, DSACM is the combination between RSA and OTP. RSA is chosen to exchange the secret key. In addition, the secret key is encrypted by using the RSA’s public key. The document is then transferred to the signer. On the signer’s side, the private key is selected to decrypt the encrypted secret key to obtain the secret key (Duong, D.H., 2020). In contrast, OTP is selected to sign the constant on the signer side and to validate the digital signature on the verifier side. Consequently, the digital signature is continuously changing, but the message remains constant. Therefore, attackers are unable to use this ambiguous digital signature. Furthermore, a benefit of DSACM is that the constant message is unchanged before the signing process. This indicates that the signature procedure is always required once. Additionally, processing time is also considered. There are several approaches proposed to minimize time without compromising security.

## 2 Related Works

In this part, the overviews of OTP, RSA, and a few applications of RSA with digital signature are discussed.

### One Time Pad

One Time Pad (OTP) is a kind of symmetric key cryptography. It differs from the other algorithms in this family in that the secret key must be implemented only once and is destroyed immediately upon completion of the decryption procedure. Although the technique behind OTP, which is an exclusive or operation, is quite simple, OTP is regarded as the perfect secrecy algorithm. Assume  $k$  is the secret key and  $m$  is the original plaintext, the encryption process is shown in equation (1)

$$c = m \oplus k \quad (1)$$

Where  $c$  is the encrypted message or the ciphertext. Furthermore,  $m$  can be recovered by using equation (2)

$$m = c \oplus k \quad (2)$$

However, bit length of  $k$  must be always equal to  $m$

### RSA

RSA is one of the most well-known public key cryptography algorithms. In 1978, Ron Rivest, Adi Shamir, and Leonard Adleman invented this algorithm. In fact, RSA derives its name from their surnames. RSA's security is based on the difficulty of factoring (Somsuk, K., 2021), (Somsuk, K., 2022), (Wu, M.E., 2014), (Pollard, J.M., 1974) and (Pollard, J.M., 1974). If the modulus is factored, RSA is entirely broken. Nevertheless, a minimum of 1024 bits of RSA (Alexander, K., 2021), (Eduardo, O.J., 2020) and (Simin, D., 2023) should be chosen so that it is extremely difficult to retrieve the encrypted message. Additionally, RSA may be utilized for both data encryption and digital signatures. However, only digital signature will be examined in this study. There are three procedures for using RSA in digital signature implementation:

#### Process 1 (Key Generation Process)

It is the procedure through which parameters for the signing and verifying procedures are generated. There are 5 steps in this process:

- Step 1: Generate two large prime numbers randomly,  $p$  and  $q$
- Step 2: Compute the modulus,  $n = p * q$
- Step 3: Compute Euler's function,  $\Phi(n) = (p - 1) * (q - 1)$
- Step 4: Select the public key ( $e$ ) from the following conditions:  $1 < e < \Phi(n)$  and  $\text{gcd}(e, \Phi(n)) = 1$
- Step 5: Compute the private key,  $d = e^{-1} \text{ mod } \Phi(n)$  where  $e^{-1}$  is modular inverse of  $e$  modulo  $n$  that can be calculated by using Extended Euclidean Algorithm (Mostafizur, R., 2009) and (Atef, I., 2016).

After completing process 1, the signer will disclose the parameters  $e$  and  $n$ . However, the remainder will be treated confidentially at signer.

#### Process 2 (Signing Process)

The signer will prepare the message to be signed by using equation (3)

$$c = m^d \text{ mod } n \quad (3)$$

Where  $m$  is the plaintext and  $c$  is the digital signature

After finishing the signing process, the signer will send the digital signature to the receiver.

**Process 3 (Verifying Process)**

The verifier will validate the digital signature using equation (4). If the answer equals  $m$ , then the digital signature is correct. In contrast, if the result is not equal to  $m$ , the system will reject the digital signature.

$$m' = c^e \text{ mod } n \tag{4}$$

From equation (4), the digital signature will be approved whenever  $m'$  equals to  $m$ .

Normally,  $d$  is always provided a large value to prevent attacks with revealed techniques such as Wiener’s attack (Wiener, M., 1990) and (Boneh, D., 2000). It has a direct impact on the signer, as modular exponentiation with a large exponent must be computed. Chinese Remainder Theorem (CRT) (Shaoqiang, B., 2008) and (Wu, C.H., 2001) was therefore devised to accelerate the signing procedure. It is proposed to divide  $d$  into two sub values and compute two modular exponentiations with low exponents. Consequently, time is undoubtedly decreased.

There are four steps to compute the digital signature by using CRT:

Step 1: Compute  $d_p$  and  $d_q$  by using equation (5) and equation (6), in order

$$d_p = d \text{ mod } p - 1 \tag{5}$$

$$d_q = d \text{ mod } q - 1 \tag{6}$$

Step 2: Compute  $c_p$  and  $c_q$  by using equation (7) and equation (8), in order

$$c_p = m^{d_p} \text{ mod } p \tag{7}$$

$$c_q = m^{d_q} \text{ mod } q \tag{8}$$

Step 3: Compute  $y_p$  and  $y_q$  by using equation (9) and equation (10), in order

$$y_p = p^{-1} \text{ mod } q \tag{9}$$

$$y_q = q^{-1} \text{ mod } p \tag{10}$$

Step 4: Compute the digital signature by using equation (11)

$$c = (c_p * y_p * q + c_q * y_q * p) \text{ mod } n \tag{11}$$

Because of low exponents in equations (5) through (11), it follows that CRT may also be used to sign digital signatures with time reduction.

**RSA Applications**

Presently, RSA is chosen by many applications to protect confidential data or authentication. In this section, examples of the application that include RSA for implementation are reviewed.

In 2010, Li Ming Xin and Kang Fend (Xin, L.M., 2010) devised a method for enhancing the Login System’s security. RSA is utilized to secure transmitted username and password from the client via an encryption mechanism. In fact, RSA parameters are produced at the server side. Accordingly, clients can encrypt their login and password utilizing an encryption procedure. After the ciphertexts reach the server, the decryption procedure may be used to retrieve the login and password. Furthermore, in this year, RSA is also selected to secure the medical images (Nadjia, A., 2010). Normally, medical information is classified information. Therefore, it must be changed into an unreadable message before transmission through an insecure channel. In addition, there are several information in the image,  $m$ . Before encrypting each sub-image, it must be divided into as many sub-images as  $m_i$ , where  $m_i < m$ .

In 2015, digital image authentication by using digital signature (Shahzad, A., 2015) was introduced. RSA has been chosen to deploy digital signature. The algorithm was tested on Lena image with Python language. Furthermore, whether outsiders may disrupt this transmission. By comparing the hashes, it is simple to validate the difference between the obtained image and the original image.

In 2020, the technique to authenticate e-certificate (Somsuk, K., 2020) was introduced. Typically, e-certificates, which are digital image files, are signed with RSA. However, almost pixels are not changed, only the participant's name in e-certificate is different from the others' files. To save time on both the signer and validator sides, only pixels carrying the participant's name will be utilized in the process.

Later, in 2020, Xiaixin Jiao et al. presented the image encryption system (Kaixin, J., 2020) to protect digital images. This scheme is from the combining between the Arnold map with RSA. In fact, RSA is chosen to generate the initial parameters of generalized Arnold map. In addition, they demonstrate that this system is efficient and has robust anti attack capabilities and key sensitivity. In truth, several studies choose RSA to protect digital images such as (Seema, V., 2021), (Faten, H.M.S.A., 2020) and (Vandana, R., 2016).

Although RSA may be adopted to use with several applications, it becomes wasteful when it is used to establish digital signatures for constant messages. In truth, the constant message may be used for a variety of tasks. For example, it may be selected to operate the electronic device linked to an IoT device such as raspberry pi and ESP8266 through the internet. The constant message will be signed on the client side before being transmitted to the device via the internet. This device will validate this message using RSA's public key when it has been received. If the result equals the constant message representing the command to activate the device, the device is activated. In contrast, the device will refuse this command. The second example is the use of an e-document. Typically, the signer must sign his or her e-document with a constant message, such as his or her name, before delivering the digital signature to the recipient. After this communication reaches the recipient, it must be validated. If the target matches the constant message, the recipient will accept this digital signature. Alternatively, it is rejected.

Nevertheless, as seen by the above examples, this strategy may be attacked without the need of integer factorization techniques. In fact, they can attack the significant information by stealing the signed message sent via internet. Assuming this ciphertext was stolen along the way, intruders may use the signature procedure to send this value to control the device. It implies that applying digital signature to control the device via internet is still not secured and time for encryption and decryption processes is also increased.

Figure 1 illustrates the application of RSA's digital Signature on an e-document. In this example, the signer has signed two identical "OUM" letters at different times. Nonetheless, signed messages are not altered. Thus, if the signer can capture the information over the communication channel, they can immediately utilize the message for authentication.

### 3 The Proposed Method

The aim of this paper is to present the improvement of digital signature algorithm to solve the problem of constant message. It is called Digital Signature Algorithm for Constant Message (DSACM). OTP is used to sign the constant message rather than public key cryptography because the key must be updated each time. Furthermore, RSA is selected to exchange OTP's secret key.

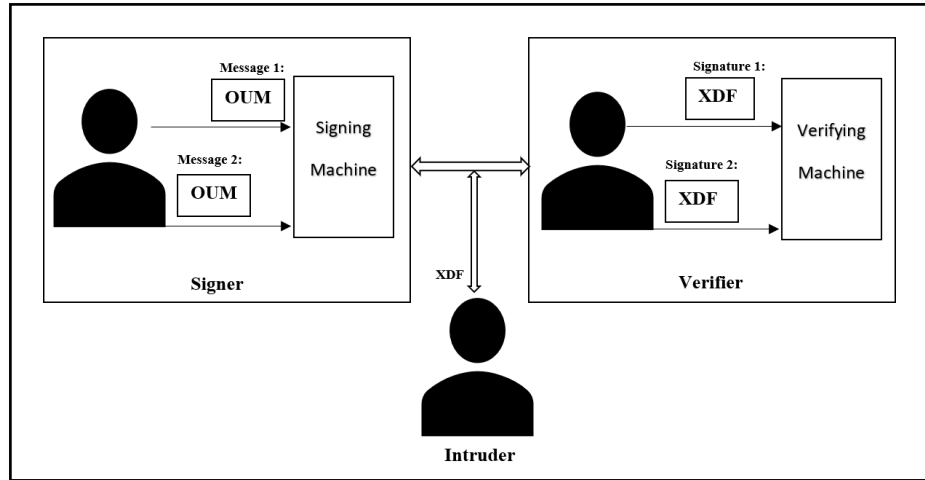


Figure 1: The Example of Applying RSA's Digital Signature with E-Document

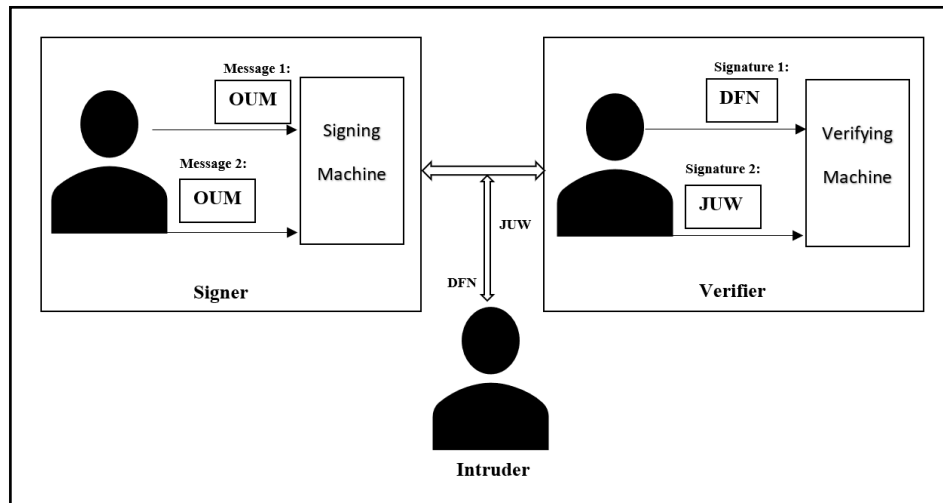


Figure 2: The Example of Applying DSACM with E-Document

Figure 2 shows an illustration of DSACM for signing and validating the digital signature of a constant message. The benefit is that the digital signature is always changing, but the message remains constant. Therefore, it is extremely challenging for attackers to obtain the secret information.

The registration method of DSACM, seen in Figure 3, allows the signer to provide the public parameters to be stored in the server's database, which is the verifying machine. Signer begins the process by generating RSA parameters and selecting a username,  $u$ , which must be unique. Later, signer will send RSA's public parameters,  $e$  and  $n$ , and  $u$  to keep in database. Normally, the server will retain the new client's parameters when  $u$  is not found in the database. In addition, prior to storing all parameters in the database, the server will produce  $m = \text{CASC}(u)$  and  $l$ , the length of  $m$ . As illustrated in Example 1,  $\text{CASC}(u)$  is the linking character during the binary representation of the ascii code for each letter of  $u$ .

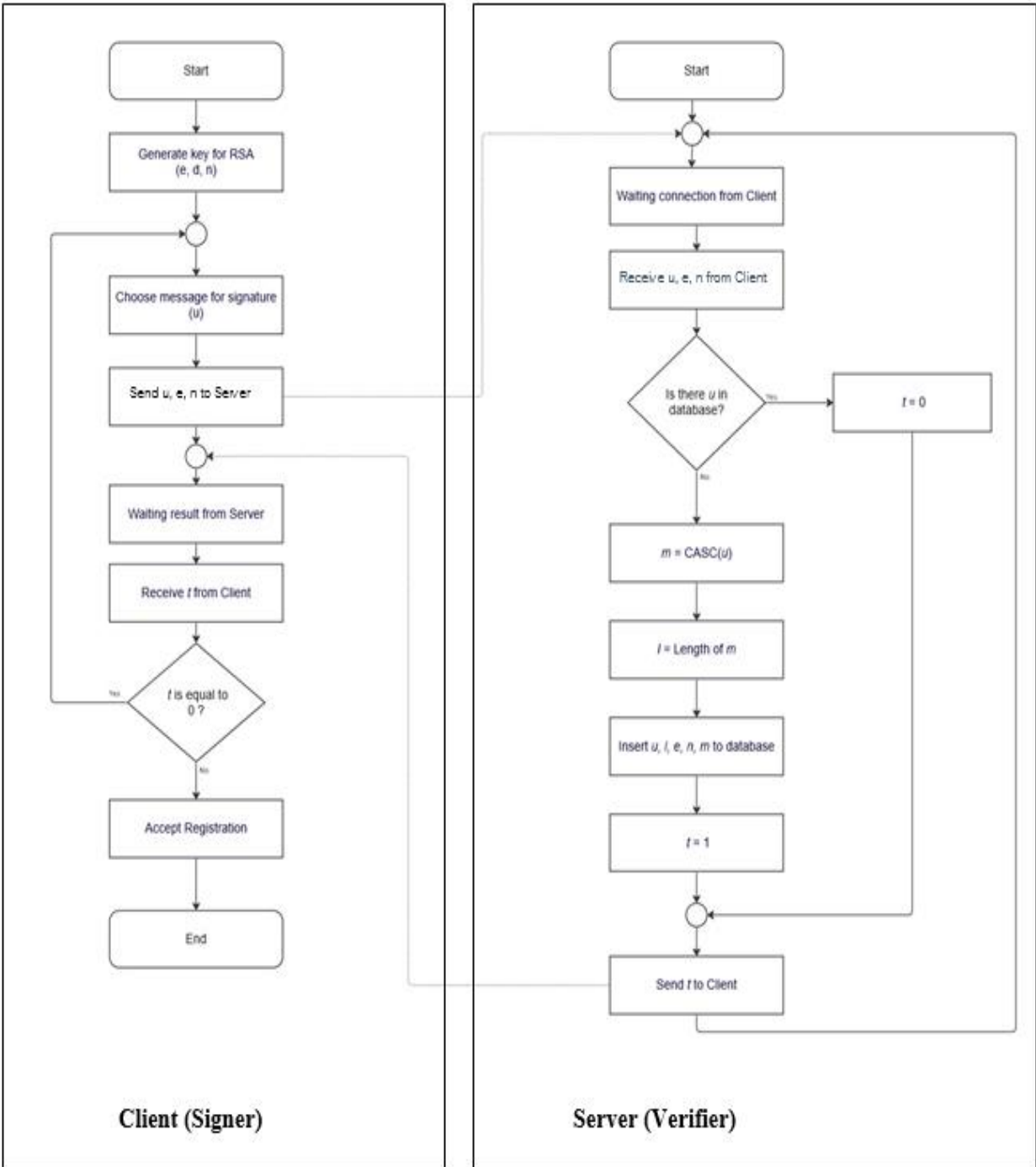


Figure 3: Algorithm of Registration System for DSACM

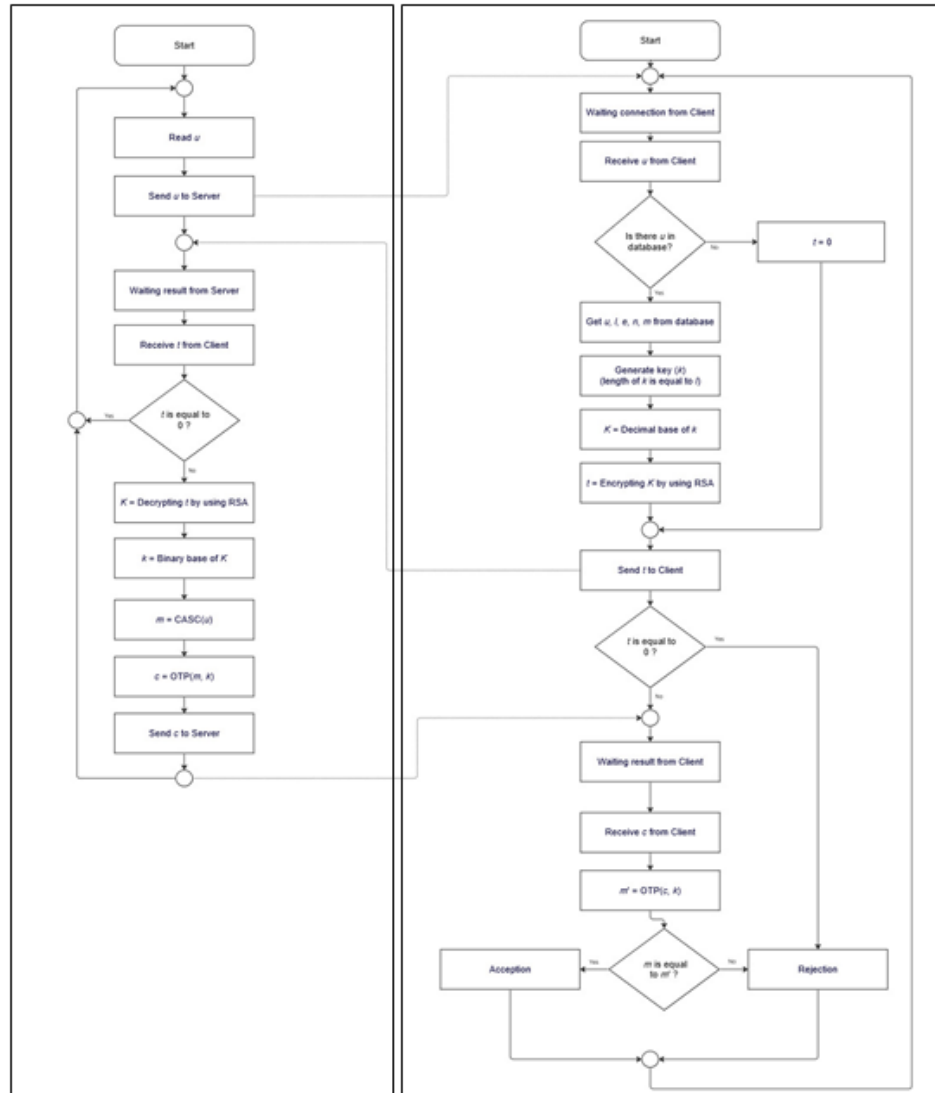


Figure 4: Algorithm of Signing and Verifying Digital Signature for DSACM

**Example 1** Assigning  $u = \text{“CAT”}$  finding  $m$  and  $l$

**Sol:** Because the binary ascii code of each alphabet of  $u$  is,

$C = 01000011$ ,  $A = 01000001$  and  $T = 01010100$

Therefore,  $m = \text{“}01000011\text{”} + \text{“}01000001\text{”} + \text{“}01010100\text{”}$

$= 010000110100000101010100$

And  $l = 24$

In fact,  $m$  and  $l$  are also keep in database with  $u$ ,  $n$ ,  $e$ .

Assuming the verifying machine has already kept client’s parameters in database, client can sign the constant message by using the DSACM algorithm in Figure 4. First, signer requests OPT’s key from the server by sending  $u$  to the server. After  $u$  arrives to the server, it must be looked up in the database to choose the other parameters for the same record. Later, server will generate OTP’s key,  $k$ , that length equals to  $l$  and convert to decimal base,  $K$ . Before sending  $K$  to signer, it must be encrypted using RSA. In practice, signer will decrypt the ciphertext to retrieve  $K$  once it reaches signer’s area. Later,  $k$  is



reconstructed by converting  $K$  to binary code. The final step on the signer side is to use  $k$  to sign message  $m$  to get  $c$  before submitting the signed message to the server for verification. After receiving  $c$ , server will decrypt this value to verify the result.

**Example 2** The implementation of DSACM for signing and verifying

**Sol:** Assuming  $n = 412338779203669$  ( $16879559 * 24428291$ ),  $e = 749$ ,  $d = 242228364050949$  and  $u = \text{"CAT"}$  ( $m = 010000110100000101010100$ )

Digital Signature#1 (Digital Signature of CAT)

Signer:

Step 1: Send  $u = \text{"CAT"}$  to Server

Server:

Step 1: Receive  $u = \text{"CAT"}$  from signer

Step 2: Select  $u, m, l, e, n$  from database where  $u = \text{"CAT"}$

Step 3:  $k = 111100010110011111001111$

Step 4:  $K = 15820751$

Step 5:  $t = 368520302498580$

Step 6: Send  $t = 368520302498580$  to signer

Signer:

Step 2: Receive  $t = 368520302498580$  from server

Step 3:  $K = 15820751$

Step 4:  $k = 111100010110011111001111$

Step 5:  $m = 010000110100000101010100$

Step 6:  $c = 101100100010011010011011$

Step 7: Send  $c = 101100100010011010011011$  to server

Server:

Step 7: Receive  $c = 101100100010011010011011$  from signer

Step 8:  $m' = 010000110100000101010100$

Step 9: Because  $m' = m$ , Accept this authentication!!

Digital Signature#2 (Digital Signature of CAT)

Signer:

Step 1: Send  $u = \text{"CAT"}$  to Server

Server:

Step 1: Receive  $u = \text{"CAT"}$  from signer

Step 2: Select  $u, m, l, e, n$  from database where  $u = \text{"CAT"}$

Step 3:  $k = 101000000100111000010101$

Step 4:  $K = 10505749$

Step 5:  $t = 300708930648149$

Step 6: Send  $t = 300708930648149$  to signer

Signer:

- Step 2: Receive  $t = 300708930648149$  from server
- Step 3:  $K = 10505749$
- Step 4:  $k = 101000000100111000010101$
- Step 5:  $m = 010000110100000101010100$
- Step 6:  $c = 111000110000111101000001$
- Step 7: Send  $c = 111000110000111101000001$  to server

Server:

- Step 7: Receive  $c = 111000110000111101000001$  from signer
- Step 8:  $m' = 010000110100000101010100$
- Step 9: Because  $m' = m$ , Accept this authentication!!

In fact, the purpose of this example is to demonstrate that all different ciphertexts are derived from the same plaintext.

However, when server transmits  $t$  to signer, it will not process anything until signer provides  $c$ . In fact, when  $c$  arrives, server will process again by computing an exclusive or operation. While the server is waiting for data from the signer,  $c'$  may be calculated as  $c' = m \oplus k$ . For the case when the server has received  $c$ , instead of comparing  $m$  and  $m'$ , it will compare  $c$  with  $c'$ . That's why the amount of time is reduced. Commonly, this method is referred as DSACMV2.

Typically, the secret key should be given as a large value to avoid being readily compromised by intruders. In addition, the class in the Java programming language known as Big Integer is chosen for implementation in this work since it may represent an unlimited datatype. Nonetheless, the time necessary to find the large integer increases while the size increases. Therefore, to reduce the time needed to create the secret key, Algorithm 1, is called GSCS, is presented to generate the secret key by generating a large number from small integers which length is stable and then connecting them to obtain the result.

Algorithm 1: GSCS
Input: $l, g$ ( $l$ is length and $g$ is the number of group members)
Output: $k$ (the secret key)
Condition: $g \mid l$
<ol style="list-style-type: none"> <li>1. <math>k \leftarrow \text{null}</math></li> <li>2. <math>r \leftarrow l/g</math></li> <li>3. For <math>i = 0</math> to <math>r - 1</math> Do</li> <li>4.     <math>t \leftarrow</math> Creating the <math>g</math> bits of the secret key randomly</li> <li>5.     <math>k \leftarrow k + t</math></li> <li>6. End For</li> </ol>

**Example 3** Generate 30 bits of the secret key using GSCS when  $g = 10$

**Sol:** In the example 3, it indicates that  $l = 50, g = 10$ , therefore

Line 1:  $k = \text{null}$

Line 2:  $r = 50/10 = 5$

Loop 1:

Line 4:  $t = 1100111101$

Line 5:  $k = 1100111101$

Loop 2:

Line 4:  $t = 1000101001$

Line 5:  $k = 1100111101\ 1000101001$

Loop 3:

Line 4:  $t = 1000000001$

Line 5:  $k = 1100111101\ 1000101001\ 1000000001$

Therefore,  $k = 0111101\ 1000101001\ 1000000001$

Algorithm 2: EOS
Input: $l$ ( $l$ is length)
Output: $k$ (the secret key)
<ol style="list-style-type: none"> <li>1. <math>k \leftarrow \text{null}</math></li> <li>2. For <math>i = 0</math> to <math>l - 1</math> Do</li> <li>3.     <math>t \leftarrow \text{Random a small integer (0 to 9)}</math></li> <li>4.     IF <math>t</math> is an even number</li> <li>5.         <math>k \leftarrow k + "0"</math></li> <li>6.     Else</li> <li>7.         <math>k \leftarrow k + "1"</math></li> <li>8.     End IF</li> <li>9. End For</li> </ol>

However, Algorithm 2, it is also called Even an Odd Selection (EOS). The value of each bit is derived from a random integer. If the integer is even, the bit is set to "0". In contrast, it will be assigned the value "1". In addition, they are all interconnected in constructing a large integer. However, EOS does not require Big Integer class for the implementation.

## 4 Results and Discussion

In this part, three different issues will be discussed. First, experimental findings on the time required to produce the OTP key on the server are presented to select the algorithm needing the lowest amount of time. Then, the best algorithm from the initial experiment will be selected for the second experiment. In fact, the second issue is the evaluation of total process duration from both the server and signer sides. In addition, CRT is applied with RSA to speed up signing process. However, the competitors include DSACM and DSACMV2. In addition, all tests were carried out on a 2.53 GHz Intel® Core i5 with 8 GB of RAM so that the same resource could be controlled. The last topic is the discussion about security level between the proposed methods (DSACM and DSACMV2) and applying public key cryptography with digital signature.

Figure 5 compares the time required to produce the OTP's key on the server. There are three competitors. The first competitor is the one-step randomization of a large integer. The other competitors are GSCS and EOS in order. However, only the first competitor and GSCS require Big integer Class for the implementation. According to the findings of the experiments, EOS is the most efficient means of generating the key for OPT. The reason is that EOS does not require the Big Integer Class.

After that EOS is chosen for considering overall time to finish process in both of server and signer sides. The experimental results in figure 6 is shown that DSACMV2 is always faster than DSACM. The reason is that both of the following processes can be calculated at the same time. The first process is RSA's decryption process and OTP's encryption which are in signer side. The second process is OTP's encryption process which is in server side. Because two procedures are necessary on the signer's side while only one is required on the server's side, both processes occur simultaneously, it follows that the

server will complete the process before the signer. In addition, DSACMV2 is five to ten percent quicker than DSACM.

The last issue in this section is to discuss about the security level that DSACMV2 is compared with the applying public key cryptography with digital signature. In fact, as noted in the section on related works, implementing public key cryptography with digital signature with a constant message is inappropriate. Nonetheless, DSACMV2 can resolve this issue. In fact, the information that intruders may trap on the communication channel is  $t$  and  $c$ . Normally,  $t$  is the ciphertext of OTP's key which is generated in server. Moreover, it is an uncertain value because it is changed every time. This other value,  $c$ , which is generated in signer side is the digital signature of the constant message. Similarly, because  $c$  is likewise an output of the OTP process, its value is also uncertain. Therefore, intruders cannot utilize from  $t$  and  $c$ .

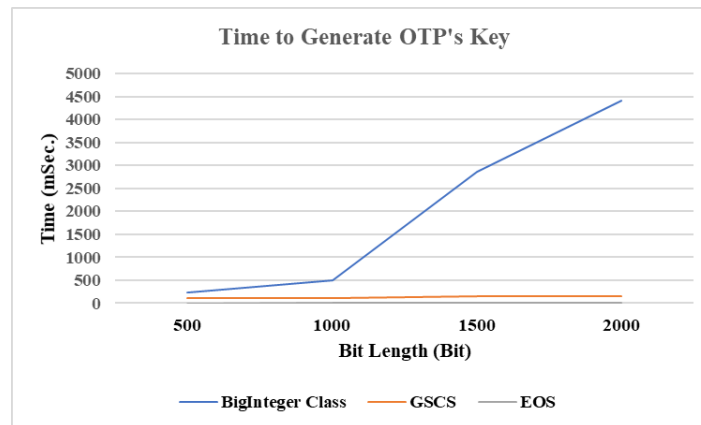


Figure 5: Time to Generate OTP's Key

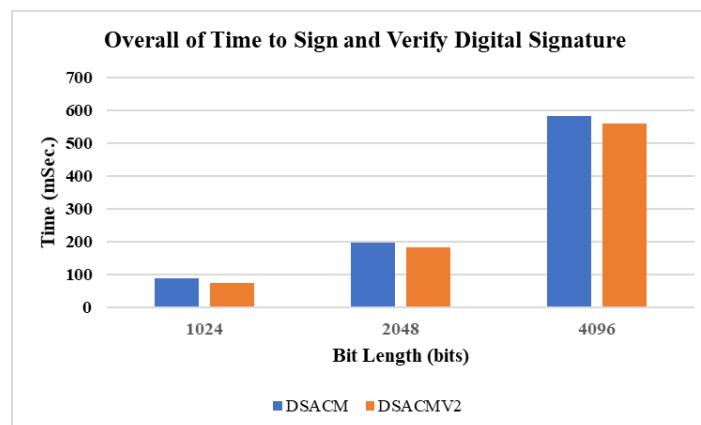


Figure 6: Overall of Time to Sign and Verify Digital Signature

In fact, DSACMV2 is superior than other methods since the constant message is always specific and just one RSA's encryption and decryption operation are needed. On the other hand, if the sum of the constant message and the integer is more than the private key, then the other methods may need additional steps to implement. Moreover, unlike other techniques, DSACMV2 does not require an algorithm to trim the constant message after the verification process is complete, as the result of the verification process is already the constant message. Figure 6 indicates that DSACMV2 completes all procedures in both the signing and verifying sides in less than 600 milliseconds. This does not include the duration that it takes to transfer data through the network channel. Furthermore, intruders must solve

the integer factoring problem and attack OPT, the perfect secrecy algorithm to attack DSACMV2 that is extremely difficult, particularly when modulus bit length is long.

## 5 Conclusion

In this work, we offer novel techniques for signing and verifying the digital signature of a constant message. Both are DSACM and DSACMV2. However, DSACMV2 is significantly better to DSACM since concurrency can be estimated for some processes. In addition, techniques to create the OTP's key are suggested, and the optimal approach is chosen for integration with the system. In fact, EOS is the best algorithm. There are two issues to consider, time and security level. For security level, it implies that DSACM and DSACMV2 can be chosen to sign and verify digital signature of the constant message and it is very hard to break these systems, because intruders must solve integer factoring problem and attack OPT. In addition, the overall time required to complete the operation on both the signer and verifier sides is less than one minute, although bits length of modulus is 4096. Therefore, DSACM and DSACMV2 are suitable to be selected to sign and verify digital signature of the constant message.

## References


- [1] Alam, S., Jamil, A., Saldhi, A., & Ahmad, M. (2015). Digital image authentication and encryption using digital signature. *In IEEE International Conference on Advances in Computer Engineering and Applications*, 332-336.
- [2] Al-Kadei, F.H.M.S., Mardan, H.A., & Minas, N.A. (2020). Speed up image encryption by using RSA algorithm. *In IEEE 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 1302-1307.
- [3] Alzubi, J.A. (2021). Blockchain-based Lamport Merkle digital signature: authentication tool in IoT healthcare. *Computer Communications*, 170, 200-208.
- [4] Anane, N., Anane, M., Bessalah, H., Issad, M., & Messaoudi, K. (2010). RSA based encryption decryption of medical images. *In IEEE 7th International Multi-Conference on Systems, Signals and Devices*, 1-4.
- [5] Aufa, F.J., & Affandi, A. (2018). Security system analysis in combination method: RSA encryption and digital signature algorithm. *In IEEE 4th International Conference on Science and Technology (ICST)*, 1-5.
- [6] Bi, S., & Gross, W.J. (2008). The mixed-radix Chinese remainder theorem and its applications to residue comparison. *IEEE Transactions on Computers*, 57(12), 1624-1632.
- [7] Boneh, D., & Durfee, G. (2000). Cryptanalysis of RSA with private key  $d$  less than  $N^{\frac{1}{4}}$ . *IEEE transactions on Information Theory*, 46(4), 1339-1349.
- [8] Du, S., & Ye, G. (2023). IWT and RSA based asymmetric image encryption algorithm. *Alexandria Engineering Journal*, 66, 979-991.
- [9] Duong, D.H., Susilo, W., & Trinh, V.C. (2020). Wildcarded Identity-Based Encryption with Constant-size Ciphertext and Secret Key. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, 11(2), 74-86.
- [10] ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4), 469-472.
- [11] Hellman, M. (1976). New directions in cryptography. *IEEE transactions on Information Theory*, 22(6), 644-654.
- [12] Ibrahim, A., Alsomani, T., & Gebali, F. (2016). New scalable digit-serial inverter over GF (2 m) for embedded applications. *In IEEE International Conference on Advances in Electrical, Electronic and Systems Engineering (ICAEEES)*, 531-534.

- [13] Jiao, K., Ye, G., Dong, Y., Huang, X., & He, J. (2020). Image encryption scheme based on a generalized Arnold map and RSA algorithm. *Security and Communication Networks*, 2020, 1-14.
- [14] Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of computation*, 48(177), 203-209.
- [15] Kulow, A., Schamberger, T., Tebelmann, L., & Sigl, G. (2021). Finding the needle in the haystack: Metrics for best trace selection in unsupervised side-channel attacks on blinded RSA. *IEEE Transactions on Information Forensics and Security*, 16, 3254-3268.
- [16] Mathe, S.E., & Boppana, L. (2017). Design and implementation of a sequential polynomial basis multiplier over GF (2 m). *KSII Transactions on Internet and Information Systems (TIIS)*, 11(5), 2680-2700.
- [17] Miller, V.S. (1986). *Use of elliptic curves in cryptography*, 417-426. Springer Berlin Heidelberg.
- [18] Ming-xin, L., & Feng, K. (2010). An improved sign-in scheme based on RSA cryptosystem. *In IEEE International Conference on Computer Application and System Modeling (ICCSM)*, 6, 35-37.
- [19] Ochoa-Jiménez, E., Rivera-Zamarripa, L., Cruz-Cortes, N., & Rodríguez-Henríquez, F. (2020). Implementation of RSA signatures on GPU and CPU architectures. *IEEE Access*, 8, 9928-9941.
- [20] Pollard, J.M. (1974). Theorems on factorization and primality testing. *In Mathematical Proceedings of the Cambridge Philosophical Society*, 76(3), 521-528. Cambridge University Press.
- [21] Pollard, J.M. (1978). Monte Carlo methods for index computation (*mod*p). *Mathematics of computation*, 32(143), 918-924.
- [22] Rahman, M., Rokon, I.R., & Rahman, M. (2009). Efficient hardware implementation of RSA cryptography. *In IEEE 3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication*, 316-319.
- [23] Rajput, V., Tiwari, S.K., & Gupta, R. (2016). An enhanced image security using improved RSA cryptography and spatial orientation tree compression method. *In IEEE International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, 327-331.
- [24] 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.
- [25] Shakiba, A. (2021). A randomized CPA-secure asymmetric-key chaotic color image encryption scheme based on the Chebyshev mappings and one-time pad. *Journal of King Saud University-Computer and Information Sciences*, 33(5), 562-571.
- [26] Shareef, F.R. (2020). A novel crypto technique based ciphertext shifting. *Egyptian Informatics Journal*, 21(2), 83-90.
- [27] Somsuk, K. (2021). The Improvement of Elliptic Curve Factorization Method to Recover RSA's Prime Factors. *Symmetry*, 13(8), 1-15.
- [28] Somsuk, K. (2022). An efficient variant of Pollard's  $p-1$  for the case that all prime factors of the  $p-1$  in B-Smooth. *Symmetry*, 14(2), 1-17.
- [29] Somsuk, K., & Thakong, M. (2020). Authentication system for e-certificate by using RSA's digital signature. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, 18(6), 2948-2955.
- [30] Vishwakarma, S., & Gupta, N.K. (2021). An efficient color image security technique for IOT using fast RSA encryption technique. *In 10th IEEE international conference on communication systems and network technologies (CSNT)*, 717-722.
- [31] Wiener, M.J. (1990). Cryptanalysis of short RSA secret exponents. *IEEE Transactions on Information theory*, 36(3), 553-558.


- [32] Wu, C.H., Hong, J.H., & Wu, C.W. (2001). RSA cryptosystem design based on the Chinese remainder theorem. *In Proceedings of the Asia and South Pacific Design automation conference*, 391-395.
- [33] Wu, M.E., Tso, R., & Sun, H.M. (2014). On the improvement of Fermat factorization using a continued fraction technique. *Future Generation Computer Systems*, 30, 162-168.
- [34] Zhang, Y., & Jia, X. (2018). The fast image encryption algorithm based on substitution and diffusion. *KSII Transactions on Internet and Information Systems (TIIS)*, 12(9), 4487-4511.
- [35] Zhou, S. (2021). A real-time one-time pad DNA-chaos image encryption algorithm based on multiple keys. *Optics & Laser Technology*, 143, 1-11.

## Authors' Biography




**Kritsanapong Somsuk**  is an associate professor at the Department of Computer and Communication Engineering, Faculty of Technology, Udon Thani Rajabhat University, Udon Thani, Thailand. He obtained his M.Eng. (Computer Engineering) from Department of Computer Engineering, Faculty of Engineering, Khon Kaen University, M.Sc. (Computer Science) from Department of Computer Science, Faculty of Science, Khon Kaen University and his Ph.D. (Computer Engineering) from Department of Computer Engineering, Faculty of Engineering, Khon Kaen University. The area of research interests includes computer security, cryptography and integer factorization algorithms. He can be contacted at E-mail: kritsanapong@udru.ac.th.




**Sarutte Atsawaraungsuk**  received the B.Sc. and M. Sc degree in Computer Science, and the Ph.D. degree in Computer Engineering from the Khon Kaen University, Thailand. Currently, he is an assistant professor at the Department of Computer Education, Faculty of Education, Udon Thani Rajabhat University, Udon Thani, Thailand. His research interests in the Machine learning, image processing and its application. He can be contacted at E-mail: sarutte@udru.ac.th.




**Chanwit Suwannapong**  received a B.Eng., M.Eng. and Ph.D. degree in Computer Engineering from Khon Kaen University, Thailand. Currently, he is an assistant professor at the Department of Computer Engineering, Faculty of Engineering, Nakhon Phanom University, Nakhon Phanom, Thailand. He has published many publications in the area of Wireless Sensor Network, Ad Hoc Networks and Smart Agriculture. He can be contacted at E-mail: schanwit@npu.ac.th.



**Suchart Khummanee**  received the B.Eng. degree in Computer Engineering from the King Mongkut's Institute of Technology Ladkrabang, the M.Sc. degree in Computer Science from the Khon Kaen University, and the Ph.D. degree in Computer Engineering from the Khon Kaen University, Thailand. He is currently a full lecturer of Computer Science at the Mahasarakham University, Thailand. He can be contacted at E-mail: suchart.k@msu.ac.th.



**Chalida Sanemueang**  obtained a bachelor of arts in German as a second language from the faculty of Humanities and Social Science, Khon Kaen University and received a master of arts in European Studies (International Disciplinary Program) from Chulalongkorn University, Thailand. Currently, she is a lecturer at Language Center, Udon Thani Rajabhat University. The research interests focus on German as a second language, English as a Medium Instruction and spoken language systems. She can be contacted at E-mail: Chalida.sa@udru.ac.th.