Linguistic Based One Time Password

Ari Moesriami Barmawi, Bambang Ari Wahyudi and Tyoriza Pristi

Informatics Graduate Program, School of Computing, Telkom University
mbarmawi@melsa.net.id,bambangari@telkomuniversity.ac.id,
tyorizapristy@gmail.com

Abstract: Due to the increasing use of information technology in daily life, the disclosure of data in cyberspace is increased. In this case, people can access data as freely as they wish. However, in some institutions, such as banks, it is necessary to secure or protect internal data accessibility. Therefore, it is necessary to secure access. Various methods have been proposed for securing bank data access, such as One Time Passwords (OTP). OTP is usually sent via public channels. One attack that might threaten the OTP is hacking it after obtaining the message that includes the OTP. Thus, if an attacker suspects that a message contains an OTP, the data can be hacked, and the attacker can use this OTP to access the bank data. To overcome this problem, Sheshasaayee et al. proposed a method that embeds the OTP into a sentence based on Glyph. After embedding the OTP, a sentence is sent to the receivers. However, using this method, the sentence can raise suspicion because there are many Glyph characters in the sentence. Therefore, this research proposed a method for embedding the OTP into a sentence without raising suspicion about the attacker. In this case, the OTP is embedded into alphabetic characters used in the words of a sentence. Based on the experiment result, the proposed method is more natural than Sheshasaayee's method since only 10% of respondents suggested that some sentences generated by the proposed method are irregular. In comparison, about 72 % of the respondents suggested that the sentence generated by Sheshasaayee's method is irregular (not natural). Finally, it can be proved that using the proposed method, the sentence that the OTP embeds is more natural than Sheshasaayee's.

Keywords: one time password; naturalness; sentence generation; embedding; extraction

1. Introduction

Mobile banking is a bank service that allows customers to conduct financial transactions using a mobile device such as a smartphone. Since mobile banking is used for payment transaction, security issue in mobile banking becomes essential [3]. A bank's customers must register their account with the bank if they intend to transact using mobile banking. In this case, the customer has to download and install the mobile banking application on the customer's smartphone, which is exclusively attached to the customer's phone number. So, only the related customer can access his/her data in the application. In other words, a customer can only access his/her data using his/her phone.

Recently, there have been several existing schemes to prevent illegal access by illegitimate customers. One of them is two factors authentication. Two factors authentication is used to ensure that the One Time Password (OTP) requires access to something a person has and something a person knows (such as a PIN) [4]. As a result, OTP avoids several shortcomings associated with traditional (static) password-based authentication.

Suppose a customer is logging into a bank application. After logging into the application and conducting the transaction, the bank sends the bank's customer an OTP. The OTP is used to authenticate the bank's customer. However, OTP is still vulnerable against social engineering attack. Recently, social engineering attacks, as discussed by Siadati et al.[5], may have a probability of 50% success against SMS-based authentication. Social engineering attacks trick the user (customer of a bank) into relaying the verification code to the attacker [5]. Balduzzi et al. have found 22 instances of such attacks using a mobile honeypot in China [6]. This attack is called Verification Code Forwarding Attack (VCFA), proposed by Siadati et al. [5]. The VCFA

Received: March 15th, 2022. Accepted: February 8th, 2023
DOI: 10.15676/ijeei.2023.15.1.1

scenario begins with a password reset done by the attacker. In this attack, the attacker initiates a password recovery request on a service provider's (Bank's) website by entering the victim's username or/and email address and chooses to receive a verification code through an SMS message. Shortly after the verification code is sent to the victim, the attacker will phish the victim to steal the verification code and complete the password reset. The attacker needs to know the victim's phone number or/and email address to phish her. For this purpose, the attacker can easily search it from public records, social networking websites, dark webs [7,8], data from leaked databases of information (such as data breach as discussed in [9]), smartphone malware [10, 11, 12], vulnerabilities in messaging apps [13, 14,15] or employ social engineering techniques [8]. Since recently, security awareness of social engineering attacks has often been carried out [16,21.], the only way to steal/obtain an OTP after conducting a password reset is by wiretapping when the service provider sends the OTP to the victim. After the attacker obtains the OTP, he can access the victim's account.

Several methods have been proposed for securing the two-factor authentication during access to the user's account when the attacker already has the victim's credentials. One of them is a method proposed by Sheshasaayee et al. [2]. For overcoming the problems, Sheshasaayee uses the steganography concept [1,18]. In Sheshasaayee's method, the OTP is embedded into a sentence. The OTP is converted into a Unicode character (glyph), and the Unicode character is used in a sentence. This sentence is sent by SMS or email to the customer.

However, Sheshasaayee's idea is still vulnerable since the glyph is very striking and can raise suspicion, especially for attackers. When attackers recognize that the message sent by a sender contains secret information, then they try to find out the hidden message. For finding the secret message, the attacker tries to find out the binary code of the glyph, then tries to guess the Unicode/ASCII code – secret pair binary code. The probability of success prediction is about 3.3 10(-9). After finding the binary code of the glyph, the attacker discovers the hidden data.

For decreasing the suspicion of a sentence, a method for embedding OTP into a sentence based on natural language is proposed. In the proposed method, the OTP number is represented in alphabetical characters. These characters are used in words of the stego sentence. Based on the experiment result, it is shown that the irregularity of sentences generated by the proposed method is perceived as natural by about 72% of respondents. In comparison, sentences generated by Sheshasaayee are only perceived as natural by about 10% of the respondents.

2. Hiding OTP Using Unicode and Frequency of Occurrence in English Words

The basic idea of this method is hiding OTP digits based on the English Language. For securing the OTP against phishing attacks, eight alphabetical characters (letters) used in the English language have to be mapped to eight Unicode characters. Those eight alphabetical characters have the highest occurrence frequency in English words. Therefore, those characters are encoded into Unicode characters, as shown in Table 1.

Table 1. Selected English alphabets for hiding process based on the occurrence frequency.

No	Letter frequency (%)	Letter symbol	Secret pair-00	Secret pair-01	Secret pair-10	Secret pair-11
			ASCII code	Unicode	Unicode	Unicode
1	8.167	a	61	251	430	FF41
2	12.7	e	65	435	212F	FF45
3	6.094	h	68	04BB	13C2	FF48
4	6.966	i	69	456	2170	FF49
5	7.507	o	006F	03BF	043E	1D0F
6	1 per line	-(hyphen)	2014	2500	2012	2013
7	1 per line	. (full stop)	002E	00B7	323	02D1
8	19.18182	(space)	00A0	2000	2003	2002

A. Embedding Process Using Sheshasaayee"s Method

Sheshasaayee's method needs secret binary pair – ASCII/Unicode map to embed one OTP number. The process for embedding one OTP number is as follows:

A.1. Converting OTP numbers into characters that represented the OTP. Suppose the OTP is 439767; then it has to be converted into a BCD number

M= 0100 0011 1001 0111 0110 0111

Then, for representing M, 24 bits binary number is necessary. One character of the sentence can be embedded with 2 bits of a secret message. Thus, if six digits OTP are used, the characters needed to embed the OTP are 12 characters. The characters used to embed the OTP are a, i, h, e, o, -, ",", "." and " ".

A.2. Suppose the sentence used for hiding the OTP is "Bank on the go with our mobile banking service. Anytime, anywhere bank", then, the character that can be embedded by the OTP are:
"a", " ", "o", " ", "e", " ", "o", " ", "i", " ", "o", "u", " ", "o", "i", "e", " ", "a", "i", " ", "e", ".", "i", " ", "e", ",","," ", "a", "e", "e", " ", "a".
Since one character of the sentence can be embedded by two bits of the encoded secret message, then M has to be divided into two bits as follows:
M = 01 00 00 11 10 01 01 11 01 10 01 11

A.3. Mapping every two bits into one character of the sentence based on Table 1. The result of the mapping is as follows:

Dec.	4	3	9	7	6	7						
Binary	01	00	00	11	10	01	01	11	01	10	01	11
Char.	"a"	" "	"o"	" "	"e"	" "	"o"	" "	"i"	" "	"o"	" "
ASCII/Unicode	0251	00a0	006f	2002	212f	2000	03bf					
	2002	0456	2003	03bf	2002							

Note: If the two bits are "00", then there are no character changes, but if it is not "00", then the character has to be changed into the multilingual glyphs character, as shown in Figure 3.1.

A.4. The sentence after embedding the OTP is as follows:

B*a*nk on the g*o* w*i*th *o*ur m*o*b*i*le banking service. Anytime, anywhere bank.

B. Extraction Process Using Sheshasaayee's Method

For extracting the OTP from the sentence sent by the bank, the following process has to be conducted.

B.1. Converting char "a", " ", "o", " ", "e", " ", "o", " ", "i", " ", "o", " " to ASCII/Unicode based on Table 1. The result is 0251, 00a0, 006f, 2002, 212f, 2000, 03bf, 2002, 0456, 2003, 03bf, 2002.

B.2. This code is converted to binary code based on Table 1, and the result is 01, 00, 00, 11, 10, 01, 01, 11, 01, 10, 01, 11.

B.3. Two binary pairs are merged and converted into a decimal number. The results in binary are 0100, 0011, 1001, 0111, 0110, 0111, and after conversion into decimal, the OTP number is 439767.

Based on the embedding result, there are problems where some characters will raise suspicion to the attacker because those characters are printed in a specific Unicode character.

3. The Proposed Method

The proposed method assumes that when there is a request for a password reset process, the service provider sends a confirmation request to the password reset request sent by the user. This request is sent to the user's phone number, and the response must be sent to the same phone number. After the user sends the confirmation for a password reset process, a sentence representing the OTP (referred to as an OTP sentence) along with several concurrent sentences to disguise the position of the sentence containing the OTP is sent to the user via his smartphone. We assume that in the registration process, the user and service provider have agreed upon the position of the OTP sentence. Furthermore, a copy of the sentence as well as the destination telephone number are stored temporarily in the service provider. The application then checks whether the sentence entered by the user and the telephone number bound to the application is the same as the sentence and the destination telephone number stored temporarily in the service provider. If they are similar, the application converts the OTP sentence into OTP. Finally, the service provider will send the user the link for inputting the OTP via the application on the victim's smartphone, and the user will input the OTP from this link.

For overcoming the problem of Sheshasaayee's Method, information hiding based on linguistic rules is introduced. The basic idea of the proposed method is embedding a secret message into characters used in the words of a sentence sent to the customer of a bank. The character that can represent the OTP number is the characters included in the ten characters with the highest frequency of use in the corpus. In this research, articles from ICT Business magazine are used. The corpus"s ten characters with the highest frequency are mapped into decimal numbers.

Furthermore, this map is used to convert the OTP to the character that can be used in a sentence. Then, the words to generate the sentence is chosen based on characters used to represent the OTP number. Finally, a sentence is generated based on the chosen words. This sentence, called a stego-sentence, is sent to the bank's customer who needs to access their account.

The proposed method consists of two main processes, embedding and extraction process. The embedding process consists of two sub-processes, pre-processing and transaction process. The extraction process is conducted by converting particular characters used in the word of a sentence (which is sent to the customer) into OTP numbers. The extraction process is done by extracting the character used to represent the OTP number from the word in the sentence. The conversion of the character into the OTP number is based on a character-number map that has been agreed upon between the bank and the customer.

A. Embedding Process

The embedding process consists of two sub-processes, pre-processing and transaction. Pre-processing is used for generating a characters-number map. It is consisted of three processes, calculating the frequencies of characters used in the corpus, character sorting based on the frequency of character occurrence in the corpus, and generating a characters-number map. The transaction process is a process for generating a sentence that will be sent to the bank's customer based on the OTP number. In this research, the character-number map is agreed upon between the bank and the customer during the registration process. In this research, the character-number map is sent to the customer via a secure channel or by conducting secret- sharing process. The overview of the embedding process is shown in Figure 3.1.

A.1. Pre-processing

The pre-processing process consists of two sub-processes: calculating the ten largest frequencies of characters used in the corpus and generating numbers for the mapping table.

- *Searching Ten Highest Frequency Characters*. The input of this process is words used in the corpus. The corpus used in this work is articles from the ICT Business magazine. The output from this process is ten characters with the highest frequencies used in the corpus, as shown in table 3.1 (a).

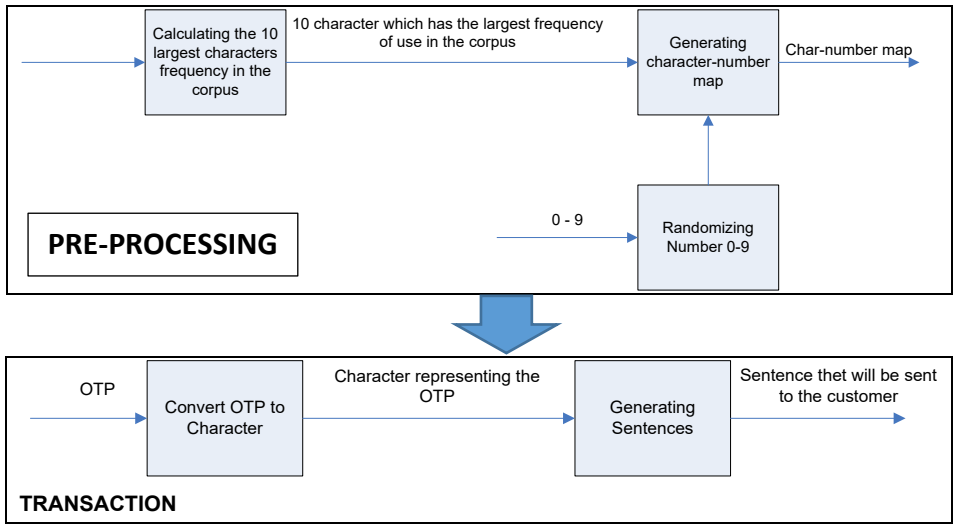

Figure 3.1 Embedding Process

- *Generating Mapping Table*

A mapping table is a table that maps OTP numbers to 10 characters with the highest frequencies of occurrences in the corpus, where these characters have to be shared between the Bank and all customers. Before generating the map, the decimal numbers 0-9 used for generating an OTP have to be randomized using a random function [20]. In this case, Mersenne Twister-based Random Number Generator from Python Library is used. The random number generator should be synchronized between the Bank and the customer so that both parties do not have to send the random number before each session starts. For example, suppose the result of decimal randomization is 4, 2, 3, 8, 6, 0, 5, 7, 1, 9. Then the mapping result is shown in Table 3.1 (b).

Table 3.1 (a) 10 characters with the highest occurrence frequency in the corpus (b) Number-Character Map.

Character	Frequency
E	1491
T	1154
N	1000
A	989
O	975
I	939
S	816
R	751
L	501
H	471

(a)

Number	Character
4	E
2	T
3	N
8	A
6	O
0	I
5	S
7	R
1	L
9	H

(b)

A.2. Transaction Process

The transaction process consists of OTP to character, generating sentences, binary labeling, generating ASCII, generating keywords, and concatenating sentences and keywords. The overview of the process is shown in Figure 3.2.

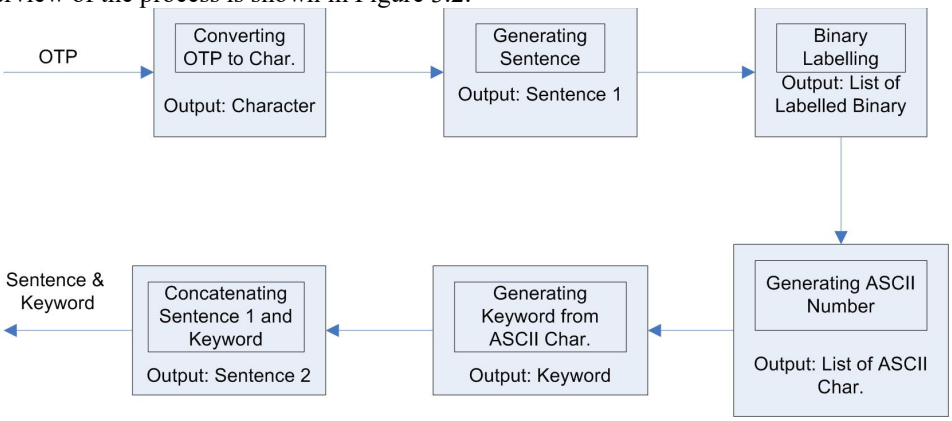

Figure 3.2 Transaction Process

- *Converting OTP Number to Character*

This process aims to convert OTP numbers into characters used in the corpus. The input of this process is the OTP number and the number-character mapping table, while the output is the character used for representing the OTP number. The OTP number is converted to a character based on the number-character mapping table.

- *Generating Sentences*

Generating sentences is conducted based on characters used to represent the OTP. This process begins by finding words in the corpus, which will be used to generate a sentence. For finding a word, a pair of words that contained two consecutive characters representing two numbers in OTP has to be searched. This process was conducted until sentences with words consisting of 6 consecutive characters representing six consecutive numbers of OTP were found. Suppose the generated sentence has a conjunction word such as “this”, “that” at the beginning. In that case, the word before the conjunction must be used. While if the conjunction word is found at the end of the generated sentences, the conjunction word must be used after the last word. These rules are determined based on English grammar [19]. After finding a sentence, the sentence length is checked to determine whether it is greater than 140 characters because the maximum length of an SMS Message is 140 characters. If its length exceeds 80 characters, then the sentence generation should be repeated until the generated sentence length is less or equal to 140 characters.

The process for generating sentences is shown in Algorithm 1.

Example:

Suppose the number-character map is {0: “i”, 1: “l”, 2: “t”, 3: “n”, 4: “e”, 5: “s”, 6: “o”, 7: “r”, 8: “a”, 9: “h”}, the OTP number is 860140, and this is the first session for the sender in sending the OTP to the recipient. Then, the OTP numbers to characters conversion based on the number-character map are [“a”, “o”, “i”, “l”, “e”, “i”].

The generating sentences start from the first character of the list of characters representing the OTP number. Since the first character is “a”, then words in the corpus that contained “a” should be collected. Suppose one of them is “meant”. Then a sentence that contained the word “meant” and a word contained “o” has to be observed. From the observation, “one” is found. Then, a sentence that contains “meant” and “one” has to be observed. In this case, “meant that one” is found. The next step is trying to find a sentence that contains “meant that one” and a word that contains “i”. In this case, “meant that one person’s ideas” is found. A similar process is conducted

until all words representing the OTP are found. Finally, the sentence resulting from the sentence generation process is “meant that one person’s ideas could reach millions”

Algorithm 1. Generating Sentences
Input:
Characters represented OTP numbers C(i) for i = {1...n} where x is OTP number index and n is the number of OTP digits; Corpus; list of conjunctions; list of prepositions

Output:
Sentence S

```
initiate S as empty
Cp ∈corpus
i ← 1
j ← 1
partitioning Sentence into words
WHILE ((the word is in conjunction or proposition list) & (it is not the end of the corpus)) DO
      Find another word in the corpus that consists C(i)
ENDWHILE
IF there is no words in the corpus consists C(i) THEN
      Generate a new OTP
ELSE
    WHILE ((i <= n) & (there is a word C(j) in the corpus) & (there is a sentence S which contain
    word C(j) and the previous sentence S) & (i<>n))   DO
                  wordpad= words between S and word C(j)
                  S= S ||wordpad || word C(j)
                  i=i+1
                  j=j+1
                  Find a word C(j) that represent C(i)
     ENDWHILE
     IF (i=n) and word C(j) is in the list of conjunction or preposition    THEN
                S=S || previous S || word C(j) || a word after word C(j)
     ELSE IF i<n THEN
                Generate a new OTP number
           ELSE return S
           ENDIF
       ENDIF
ENDIF
```

- *Binary Labeling*
The binary labeling process aims to mark characters in a word used to represent the OTP number. The label consists of 5 bits. The first bit marks the word that consists of the character representing the OTP number. Label "1" is given to a word containing a character representing the OTP number. Otherwise, it is only labeled by 1 bit "0". The following 4 bits is the label to mark the character’s position in the related word. If the word was not used, then 4 bits label is not used.

- *Generating ASCII Characters*
After the labeling process, the binary labels are converted into ASCII code. For converting the binary number into ASCII code, it has to be divided into 5 bits. Furthermore, every 5 bits of binary number should be converted into ASCII code. If the length of the binary number is less than 5 bits, then bit “0” has to be inserted such that the label will be 5 bits. Finally, “011” padding bit is inserted in front of the 5 bits label.

- *Generating Keyword*

Generating keywords is used to find words that contain characters resulting from the binary to ASCII label converting process. These keywords have to be stored in the metadata.

Generating a keyword starts with finding a word in English Dictionary containing ASCII label. Since not all ASCII characters can be mapped into Alphabetic characters, then two ASCII codes can be mapped to one alphabetic character. Thus, there is repetition after a specific value. Based on the ASCII table, the Alphabetic character starts from 65, representing A, and the end of the alphabetic character Z is represented by 90. Then, after 90, the ASCII has to be the map to the Alphabetic character used by other ASCII codes. For example, 91 has to be mapped to B, where the ASCII code of B is 66. For distinguishing the main character and the repetition character, a "*" mark is inserted before the alphabetic character. Furthermore, 91 ASCII Code is mapped to "*B". The detailed map is shown in Table 3.2.

Table 3.2 Modified ASCII Table

N0	HEXA (ORIGINAL ASCII)	FORMULA {ASCII < 65, formula = 65+64-HEXA ORIGINAL ASCII} {ASCII >90, formula = 65+HEXA ORIGINAL ASCII-90 } {ASCII 65-90 still used the original ASCII}
1	64 (@)	65 (*A)
2	65 (A)	A
3	66 (B)	B
4	67 (C)	C
5	68 (D)	D
6	69 (E)	E
7	70 (F)	F
8	71 (G)	G
9	72 (H)	H
10	73 (I)	I
11	74 (J)	J
12	75 (K)	K
13	76 (L)	L
14	77 (M)	M
15	78 (N)	N
16	79 (O)	O
17	80 (P)	P
18	81 (Q)	Q
19	82 (R)	R
20	83 (S)	S
21	84 (T)	T
22	85 (U)	U
23	86 (V)	V
24	87 (W)	W
25	88 (X)	X
26	89 (Y)	Y
27	90 (Z)	Z
28	91 ([)	66 (*B)
29	92 (\)	67 (*C)
30	93 (])	68 (*D)
31	94 (^)	69 (*E)
32	95 (_)	70 (*F)

Based on Table 3.2, the word selection is conducted using Rule 1 or Rule 2. Rule 1 is conducted for determining character position, and Rule 2 is conducted for generating marks.

The keyword selection starts by checking the number of characters, whether odd or even. The selection process is started from the end of ASCII characters. If the number of characters represents the odd label, then keywords were generated based on the rule 1 number 5. Afterward, keyword selection is started by converting the last two characters into the keyword. If the number of characters representing the label is even, keyword selection is started by following rule 1

number 1 until 4. Then, it is repeated until all labels have been converted to keywords. The beginning and end of each word were used to represent two characters (representing the labels).

a. Rule 1

As discussed, Rule 1 is conducted for word selection and determining character position. Rule 1 consisted of the following rules.

1. If the alphabetic character has no sign star ("*") mark, then it is a common word, and a mark "," is inserted after the keyword.
2. If the alphabetic character has a sign "*" and the character's position is at the beginning of the keyword, then a mark ":" is inserted after the keyword.
3. If the alphabetic character has a sign "*" and the character's position is at the end of the keyword, then a mark ";" is inserted after the keyword.
4. If two alphabetic characters have a sign "*" and the positions are at the beginning and the end of a keyword, then a mark "." is inserted after the keyword.
5. Suppose the number of characters representing the label is odd. In that case, the keyword representing the last character is located at the beginning of the keywords. Furthermore, a mark " " is inserted after this keyword. Suppose the last character representing the label is "p", and the keyword representing "p" is "papyrus", then the keyword is "papyrus" (see example in this section).

b. Rule 2

Rule 2 is used to camouflage the marks when the sender needs to send the OTP to the receiver more than once. In this case, if the sender sends the OTP n times, then n sessions are necessary. Thus, the sessions are Session 1, Session 2, …, and Session n. If n is greater than 5, then the rule for session l (where l=n mod 5) is used for word selection and determining character position. Rule 2 consisted of the following rules:

1. Session 1: sign position of each keyword was not shifted.
2. Session 2: sign position of each keyword was shifted once to the right.
3. Session 3: sign position of each keyword was shifted twice to the right.
4. Session 4: sign position of each keyword was shifted three times to the right.
5. Session 5: sign position of each keyword was shifted four times to the right.

After generating the keyword, the session number was observed, and inserted symbols based on rule 2 at the end of each keyword. The process is shown in Algorithm 2.
Notes: a sign star ("*") marks an ASCII character, not in the alphabet's ASCII list. This symbol is mapped to the character starting from the beginning of the alphabet ASCII list (*A) until (*F). The table of the ASCII list is shown in Table 3.2.

After the sentence is generated, the labeling process is conducted by marking the position with the label "1" for a word that contains the character representing the OTP number. The label represents the character position in a word by 4 bits. In the case of "meant that one person's ideas could reach millions", the word "meant" contained "a", which is the character representing the first OTP number "8". Thus, the word "meant" is labeled by "1" and the position of "a" in the word "meant" is 3. Thus, the label of the character position in the word "meant" is "0011". Based on this condition, the label of the word "meant" is "10011". A similar process is conducted for each word in the sentence: "that, one, person's, ideas, could, reach, and millions". Since the word "that" does not contain a character representing the OTP number, this word is labeled by "0". Finally, the result of the labeling process is "10011 0 10001 0 10001 10100 10010 10010".

Algorithm 2. Generating Keyword
Input: ASCII-Alphabet character char(n), n is number of characters representing the binary labels
Output: Keyword

```
Keyword=" "
IF (n is odd) THEN
        (find word(char(n)-end))
        Keyword=Keyword||word(char(n)-end)|" "
        n=n-1
ENDIF
WHILE n>0 DO
      (find word(char(n)-begin, char(n-1)-end))
      CASE of
         char(n) and char (n-1) has no sign "*": word(char(n)-begin, char(n-1)-end) =
                                                  word(char(n)-begin, char(n-1)-end)|| ","
         char(n) has sign "*" and char (n-1) has no sign "*": word(char(n)-begin, char(n-1)-end)
                                                  =word(char(n)-begin, char(n-1)-end) ||":"
         char(n) has no sign "*" and char (n-1) has sign "*": word(char(n)-begin, char(n-1)-end)
                                                  = word(char(n)-begin, char(n-1)-end)||";"
         char(n) and char (n-1) each has sign "*": word(char(n)-begin, char(n-1)-end)
                                                   =word(char(n)-begin,char(n-1)-end)|| "."
       ENDCASE
       Keyword=Keyword||word(char(n)-begin, char(n-1)-end)
       n=n-2
ENDWHILE
RETURN (Keyword)
```

After the labeling process, keyword generation is conducted. Suppose this is the first time the sender sends the OTP to the receiver. This process begins by converting binary labels to ASCII characters. Converting binary labels to ASCII characters is started by dividing the array of binary labels into 5 bits groups. If the length of the binary number was less than 5 bits, then bit "0" is padded after the last bit of the binary label. Then, "010" is padded in front of the 5 bits such that the binary array for the word "meant" is "01010011". This binary number is further converted into ASCII characters. In this case, the ASCII character resulting from the label is "s". This process is conducted for all words in the sentence. Finally, ASCII characters represented the label" 10011 0 10001 0 10001 10100 10010 10010" are "s h t m d t p"

For generating a keyword from "h t m d t p", the number of characters representing the label is observed. Since the number of characters representing the label is odd, rule 1 number 5 is implemented on the last character, "p". Afterward, the character "p" is embedded into a word in English Dictionary containing the character "p" at the beginning of the word, which is "papyrus", and inserted mark "space" after the word to mark that the number of characters is odd. In this case, the word is "papyrus". The next step is converting "t" and "d" into a word. Since the characters do not contain "*", rule 1 number 1 is applied, and the mark "," is inserted after the word "tripod". Repeat these steps until all characters have been converted into words. The process result is "papyrus tripod, tantrum, harness,". These keywords are stored in the metadata of the text.

B. Extraction Process

The extraction process is conducted by the application installed in the receiver's phone. First, the receiver extracts data from the sender by converting the data to the OTP number. The extraction process consists of four processes, i.e., Keyword to ASCII character conversion,

ASCII Binary conversion, sentence to alphabetic character conversion, and character to OTP number conversion. The overview of this process is shown in Figure 3.3.

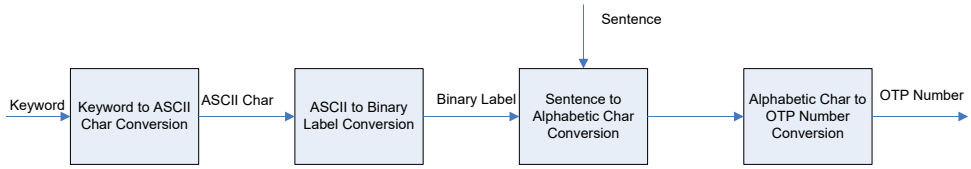

Figure 3.3 Extraction Process

B.1. Keyword to ASCII Character Conversion

In this process, the keywords are converted into characters representing the labels. This process was conducted to observe whether there is a space after the first keyword. If there is a space, then the number of characters representing the labels is odd. Otherwise, it is even. After observing the space, the session number should be observed to relocate the symbols after the keywords. Furthermore, the symbol after each keyword should be observed for determining the ASCII Characters type (whether the character is *-characters or not) representing the labels. At the end of this process, all keywords are converted into ASCII Characters representing the labels. The process is shown in Algorithm 3.

Algorithm 3. Converting Keyword to ASCII Character

```
Input: Keyword, m=number of words in Keyword
Output: ASCII Character
j=1
n=1
If (there is a " " after the first word)
THEN   ASCII-char(n)=First-char(word(j))
       n=n+1
       j=j+1
ENDIF
WHILE j<= m
       CASE of
              Mark after word(j) is "," : ASCII-char(n)= First-char(word(j)
                                          ASCII-char(n+1)=Last-char(word(j)
              Mark after word(j) is ":" : ASCII-char(n)= *First-char(word(j)
                                          ASCII-char(n+1)=Last-char(word(j)
              Mark after word(j) is ";" : ASCII-char(n)= First-char(word(j)
                                          ASCII-char(n+1)= *Last-char(word(j)
              Mark after word(j) is "." : ASCII-char(n)= *First-char(word(j)
                                          ASCII-char(n+1)= *Last-char(word(j)
       ENDCASE
       j=j+1
       n=n+2
ENDWHILE
RETURN ASCII-char (l) for l=1,n
```

B.2. Keyword to ASCII Character Conversion

For obtaining OTP, the ASCII character has to be converted to the binary label based on the ASCII table, as shown in Table 3.2. After obtaining the binary label, a process for removing the padding bits is conducted. This process removed the padding bit "010" from each bit array representing the ASCII character. After removing the padding bit, the first bit of the label has to be observed. If the first bit of the label is 1, then it means that the word contains a character representing the OTP number. Then, observe the character's position in the related word

represented by the following four bits of the binary label. If the first bit is 0, then the word does not contain any character representing the OTP number. This process is conducted until the end of the binary label.

Example:
As discussed in the example of the embedding process, the sentence resulting from this process is: “meant that one person’s ideas could reach millions# papyrus tripod, tantrum, harness”.
The extraction process begins by observing the sender’s keyword and the session number. Since, in this example, it is the first time the sender communicates with the user, then there is no mark position shifting. After observing the session number, the mark of the first word ("papyrus ”) has to be observed using rule 5. Since there is a " " after the word papyrus, then the number of ASCII characters representing the label is odd. Therefore, the first keyword is "papyrus", such that the ASCII character is the first character of the word "papyrus", which is "p". The next word is "tripod,". Mark comma (,) means that rule 1 is used to convert the word "tripod" to the ASCII character. Thus, the characters in the keyword “tripod” used to represent the ASCII characters are "t" and "d". Repeat this process until the last word of the keyword. This process result is “s h t m d t p”.

The binary label resulted by converting ASCII characters to binary numbers based on Table 3.2 and removing padding “010” from the binary label is 10011 0 10001 0 10001 10100 10010 10010.
The following process is converting the binary label into the Alphabetic character representing the OTP number. In this case, if the label of the word is 1, then the character’s position in the word is represented by the following four bits of the binary label. Since the first word "meant" has label 10011, where the first bit is 1, then it means that this word contains a character representing the OTP number. The following four bits represent the character’s position that represents the OTP number in the word "meant". Based on the label, "a" is the character that represents the OTP number because the value of the following four bits is 3. This process is repeated until the end of the label. Furthermore, the result obtained from this process is : ["a", "o", "i", "l", "e". "i"]. Finally, the OTP number 8,6,0,1,4,0 is obtained after converting the character representing the OTP number to the OTP.

4. Experiment

This section discussed the evaluation in terms of the suspicion level of the proposed method compared with Sheshasaayee’s one, along with the time complexity of the proposed method and the security analysis. The suspicion level is evaluated based on the naturalness or the irregularity of the sentence representing the OTP number.

A. Irregularity Evaluation

The suspicion level evaluation of the sentence resulting from the proposed method is conducted by comparing the suspicion level of Sheshasaayee and the proposed method. The same OTP number is used for this comparison, and the irregularity of sentences resulting from these two methods is compared.

The evaluation of the sentences resulting from those two methods is compared based on sentence irregularity. Sentence irregularity consists of two aspects. The first one is the irregularity in characters used in the sentence, and the second is the irregularity in sentence grammar, such as no subject, predicate, etcetera. If there is no irregularity caused by the characters used in the sentence, then the irregularity from the grammar point of view has to be evaluated. Suppose the sentence is irregular from the grammar point of view. In that case, the sentence has to be observed whether it has no subject, predicate, object, subject-predicate, subject-object, predicate-object, or subject-predicate-object.

Based on the 30 OTP experiments, the result of irregularity evaluated by general respondents is shown in Table 4.1. Based on Table 4.1, it can be concluded that the percentage of the irregular

sentence when using the previous method is 86.7%, while when using the proposed method is 13.3%. Based on the experiment result, it is shown that the sentences that resulted when using the previous method are less natural than the sentences that resulted when using the proposed method. Stego-sentence resulted from the proposed and Sheshasaayee's method is shown in Table 4.1.

Table 4.1 An example of a stego-sentence resulted from Sheshasaayee's and the proposed method.

NO.	OTP	Stego Sentence based on Sheshasaayee's method	Irregularity	Stego Sentence based on the Proposed Method	Irregularity
1	860140	m ant that one person's ideas could reach millions	Irregular	meant that one person's ideas could reach millions	Regular
2	511357	it departments ar e exper i encing tremendous changes as their roles expand	Irregular	it departments are experiencing tremendous changes as their roles expand	Regular
3	649819	on th i nk to understand when managing the growing pains	Irregular	one thing to understand when managing the growing pains	Regular
4	606595	mayb they just used them to play games or type	Irregular	maybe they just used them to play games or type	Regular
5	998691	i nteract w i th th growing digital world without compromising the security	Irregular	interact with the growing digital world without compromising the security	Regular
6	400329	perhaps our job for the next thirty years is to make the web available	Irregular	perhaps our job for the next thirty years is to make the web available	Regular
7	784237	anyt h i ng to prot ct our d a ta? or should we just accept that	Irregular	anything to protect our data? or should we just accept that	Regular
8	689160	advanc s i n t ech nology made by china present an opportunity	Irregular	advances in technology made by china present an opportunity	Regular
9	598460	compan i es and onlin platforms use this 'footprint' to track exactly what	Irregular	companies and online platforms use this 'footprint' to track exactly what	Regular
10	147719	internet was dev e loped in the early 1970s by vint cerf	Irregular	internet was developed in the early 1970s by vint cerf	Regular
11	160119	knowing so much about you gives online platforms and companies a lot of power	Regular	knowing so much about you gives online platforms and companies a lot of power	Regular
12	717001	lig h ting in accordance with vehicle/pedestrian movement and weather	Irregular	lighting in accordance with vehicle/pedestrian movement and weather	Regular
13	928556	can we do anything to protect our data? or should we just accept that	Irregular	can we do anything to protect our data? or should we just accept that	Regular
14	780289	onl i n platforms us th i s 'footprint' to track exactly what we are	Irregular	online platforms use this 'footprint' to track exactly what we are	Irregular
15	432011	much about you gives online platforms and companies a lot of power and a lot	Irregular	much about you gives online platforms and companies a lot of power and a lot	Irregular
16	318484	th i rty years is to make the web available to the other half of the world	Irregular	thirty years is to make the web available to the other half of the world	Regular
17	906825	w i th the cmo and ceo will also be key in identifying	Irregular	with the cmo and ceo will also be key in identifying	Regular
18	662638	it to deliver a competitive advantage as much as they do for a marketing	Irregular	it to deliver a competitive advantage as much as they do for a marketing	Irregular
19	534765	smart cit i es ar e bu i lt on the backbone of solid infrastructure	Irregular	smart cities are built on the backbone of solid infrastructure	Regular
20	591544	to m that this shift is directly related to "the internet of everything"	Irregular	to me that this shift is directly related to "the internet of everything"	Regular
21	505908	one thing to understand when managing the growing pains of it	Regular	one thing to understand when managing the growing pains of it	Regular
22	622265	intersect i on th se areas suggests an increasingly prominent	Irregular	intersection of these areas suggests an increasingly prominent	Regular
23	483953	consid r how you'll construct a workforce and culture that drives	Irregular	consider how you'll construct a workforce and culture that drives	Regular
24	557730	our personal informat i on can be collected and stored	Irregular	our personal information can be collected and stored	Regular
25	775841	trans i tion from landlines to mobile telephony is a case	Irregular	transition from landlines to mobile telephony is a case	Regular
26	437560	telephone meant that one person could connect with one	Irregular	telephone meant that one person could connect with one	Regular
27	154940	departements are experiencing tremendous changes as their roles expand to impact	Irregular	departments are experiencing tremendous changes as their roles expand to impact	regular
28	558309	worker ar na just facil i tating business goals they driving	Irregular	workers arena just facilitating business goals they driving	Irregular
29	560306	the i ntersection of these areas suggests an increasingly prominent	Irregular	the intersection of these areas suggests an increasingly prominent	Regular
30	628639	compani s and onlin e platforms use this 'footprint'	Irregular	companies and online platforms use this 'footprint'	Regular

It is proven that the naturalness of the sentences resulting from using the proposed method is better than the previous one because, in the proposed method, the sentences are generated based on a corpus and the relations between one word and other words in the corpus. Another main

difference between the previous and the proposed method is that the sentences are generated without using glyph symbols.

B. Security Analysis

Security analysis is done by calculating the probability of guessing the message (in this case, OTP) inserted in the sentence. The probability of a successful attack to obtain the OTP depends on the probability of obtaining the label indicating the location of the character representing the OTP in the sentence and the probability of successfully obtaining the number-character map guessing.

Since for obtaining the label, the attacker has to find the character that represents the label, the probability of obtaining the label is equal to the probability of obtaining the character in the keyword that represents the label. For obtaining the character that represents the label, the attacker has to know the session number, the combination of rule 1 used to generate the words, and the number-character map. Since only five possible sessions can be chosen based on Rule 2, then the probability of guessing the session number is 1/5. The probability of Rule 1 combinations used to generate the words is equal to $\left(1/_{C_2^5}\right)^n$, where n is the number of keyword and C_2^5 is the number of Rule 1 combination used in one key word. The probability to succeed the number-character map guessing is equal to $\left(1/_{C_{10}^{26}}\right)(1/10!)$. Thus, the probability for obtaining OTP is equal to

$$P(OTP) = (1/5)\left(1/_{C_2^5}\right)^n \left(1/_{C_{10}^{26}}\right)(1/10!) \quad (4.1)$$

Based on equation (4.1), the maximum probability is obtained when the number of keywords is minimum. In this case, the minimum number of words is achieved if each keyword contains two characters representing the label. In contrast, the number of keywords is minimum in the case that each word in the sentence contains one character representing the OTP number. Finally, the maximum probability of obtaining the OTP number is as follows.

$$P(OTP)_{max} = (1/5)\left(1/_{C_2^5}\right)^3 \left(1/_{C_{10}^{26}}\right)\left(\frac{1}{10!}\right) \approx 1/(9.6.10^{16})$$

In other words, the probability of obtaining OTP using the proposed method is $\leq 9.6.10^{-16}$.

In the previous method, alphabet char and some punctuations ".", ",", "- (hyphen)", "?", "!", and " " are used, and these characters can be written in 4 types of code which are ASCII and 3 Unicodes. It means that the probability of obtaining the OTP number is $\left(1/_{C_8^{33}}\right)\left(\frac{1}{4!}\right) \approx \left(1/_{3.3\ 10^9}\right) \approx (3.3\ 10^{-9})$, which is greater than the probability of obtaining the OTP using the proposed method.

C. Performance Analysis

The time complexity is used to analyze the proposed method's performance. For embedding the OTP into a sentence, the proposed method conducted sentence and keyword generations and mapped OTP to letters that are the basic of word generation. This word is further used to generate a sentence. The time complexity for the embedding process using the proposed method is equal to the time complexity of the mapping process, added by the time complexity of the sentence generation and the time complexity of the keyword generation. The time complexity of the embedding process can be calculated as $(O(n)+O(m)+(O(l) \approx O(\max(n,m,l))$, where n is the number map codes, m is the number of unique words in the corpus, and l is the number of words in English Dictionary or corpus used for generating keywords.

Meanwhile, the time complexity for the embedding process using Sheshasaayee's method only depends on the mapping process' time complexity, which equals O(n). In this case, it is shown that the time complexity for the proposed method's embedding process is greater than Sheshasaayee's. Since the number of unique words in the corpus (in our case, the corpus is the English dictionary) used for generating the keywords is larger than the number of codes in the mapping process.
The complexity of the extraction process of the proposed method depended on the number of codes in the mapping table, which is similar to Sheshasaayee's method. In this case, the time complexity equals O(n).

5. Conclusion

This research proposes an improvement of the text-based one-time password proposed by Sheshasaayee et al. [1]. Instead of text, the proposed method uses linguistic-based steganography to reduce the suspicion level of OTP embedded into a sentence. Based on the experiments and analysis, it can be concluded that the suspicion level using the proposed method is less than the previous one. This condition occurred because the sentences generated by the proposed method are less irregular than when using the previous method by the respondents.
Based on the security analysis, it is shown that the security of the proposed method is better than the previous one since the probability of obtaining the OTP is greater when using the previous method than the proposed one.

6. References

[1]. Joseph, A and Sundaram, V. (2011). Cryptography and steganography–a survey. *International Journal on Computer Technology Application*, 2(3), 626–630.
[2]. Sheshasaayee, A and Sumathy, D. (2015). Text Steganography in SMS Using Similarity of Glyphs in Unicode Characters, *Indian Journal of Science and Technology, 8(29)*, 1–6.
[3]. Bećirović, S., Bajramović, Dž. & Ahmatović, A. (2011). The role of Mobile Banking in Enhancing Economic Development. In *International Conference: Communication and Business Sector*, 89-98.
[4]. Lu, Z. and Yu, HuaZhang. (2012). One Time Password Generating Method and Apparatus, *US Patent No.US8184872*.
[5]. Siadati, H., Nguyen, T., Gupta, P., Jakobson, M. and Memon, N. (2017). Mind your SMSes: Mitigating Social Engineering in Second Factor Authentication. *Computers & Security,Volume 65,* 14-28.
[6]. Balduzzi, M., Gupta, P., Gu, L., Gao, D., and Ahamad, M. (2016). Mobipot: Understanding mobile telephony threats with honeycards. Proceedings of the 11th ACM SIGSAC Symposium on Information, Computer and Communications Security, ASIA CCS "16, New York, NY, USA.
[7]. https://www.sciencedirect.com/science/article/pii/S016740481630116X,easydmarc.com /blog/how-do-phishing-scammers-get-your-email-address/
[8]. Jakobson, M., (2016). Understanding Social Engineering Based Scams. Springer Verlag.
[9]. L.A. Times. Anthem hack exposes data on 80 million; experts warn of identity theft. http://www.latimes.com/business/la-fi-anthem-hacked 20150204-story.html#page=1, 2015. Accessed: 22-May-2016.
[10]. Kaspersky. Asacub android trojan: From information stealing to financial fraud. http://www.kaspersky.com/about/news/virus/2016/Asacub Android-Trojan-From-Information-Stealing-to-Financial Fraud, 2016. Accessed: 22-May-2016.
[11]. Symantec. Android.ackposts. https://www.symantec.com/security_response/writeup.jsp?docid=2012-072302-3943-99, 2012. Accessed: 22-May-2016.
[12]. Versprite. Android infostealer - godwon - analysis. http: //versprite.com/og/android-infostealer-godwon-analysis/, 2012. Accessed: 22-May-2016.
[13]. E. Kim, K. Park, H. Kim, and J. Song. I've got your number. In Information Security Applications, pages 55–67. Springer, 2014.

[14]. S. Kurowski. Using a whatsapp vulnerability for profiling individuals. Open Identity Summit, GI-Edition-Lecture Notes in Informatics (LNI)-Proceedings, 237:140–146, 2014.
[15]. S. Gupta, P. Gupta, M. Ahamad, and P. Kumaraguru. Abusing phone numbers and cross-application features for crafting targeted attacks. arXiv preprint arXiv:1512.07330, 2015
[16]. https://www.sciencedirect.com/science/article/abs/pii/S0045790621004912 S. Breznitz. Cry wolf: The psychology of false alarms. Psychology Press, 2013.
[17]. Mulliner, C., Borgaonkar, R., Stewin, P., and Jean-Pierre Seifert, J. (2013). SMS-based One-Time Passwords: Attacks and Defense. *DIMVA*. *LNCS.* Springer-Verlag 150–159.
[18]. Sheshaasaaye, A and Sumathy, D. (2013). A Systematic Approach towards the techniques of Text Steganography. *Proceedings of International Conference on Research Conference on Research Trends in Computer Science*. 88–96.
[19]. Ptrick Griffiths. (2016). An Introduction to English Semantics and Pragmatics, *Edinburgh University Press*.
[20]. Bhattacharjee, K., Maity, K., Dasa, S. (2018). A Search for Good Pseudo-random Number Generators: Survey and Empirical Studies. *https://arxiv.org/abs/1811.04035*
[21]. Amrut Kajave, Shazny Ahmed Hussain Nismy. (2022). How Cyber Criminal Use Social Engineering to Target Organizations. arXiv:2212.12309.

Ari Moesriami Barmawi received B. Sc from the Department of Electrical Engineering, Bandung Institute of Technology, Indonesia in 1985, M. Sc and Ph.D from the Department of Computer Science, Keio University, Japan in 1997 and 2001 respectively. Her research interests are Cryptography, Steganography, and Artificial Intelligence. She is a member of IACR (International Association of Cryptography Researcher), and ACM. Now she is an Associate Professor in Cybersecurity and Digital Forensic study program of Telkom University.

Bambang Ari Wahyudi is a Lecturer at the School of Computing of the Telkom University. Bambang graduated from Information Technology of Maranatha Christian University (2008). He received a master's degree in Informatics Engineering from the Bandung Institute of Technology (2011). He joined the Intelligent System Laboratory, School of Computing of Telkom University. His research interest includes data hiding and machine learning.

Tyoriza Pristy Saputri was born in 1994 at Cirebon City, West Java and complete her bachelor's degree in Computer Sains Education in 2015 from Indonesia University of Education (UPI), Indonesia. In 2019, She received her Master's Degree from Informatics Engineering at Telkom University Bandung. She is working as Government Imployee become an Informatics teacher at Ministry of Religion. Her research interest is in Linguistic-based Steganography for mobile banking.