Implementation of Cryptography Merkle-Hellman Knapsack Combination, Discrete Algorithm and ASCII Table Modification for PHP Source Code Security

C Wadisman$^{1,2}$, I Nozomi$^{1,3}$ and S Rahmawati$^{1,4}$

$^1$ Putra Indonesia University YPTK Padang, Indonesia
Email: cendra_wadisman@upiyptk.ac.id$^2$, irohito_nozomi@upiyptk.ac.id$^3$, sri rahmawati@upiyptk.ac.id$^4$

Abstract. The PHP programming language is now widely used by agencies and companies around the world. To distribute applications made using PHP must use the source code so that it is prone to theft of works. If the source code is found by the programmer, it will be a big problem because it can indirectly find a hole in the security of a system from the source code. To solve this problem, the source code was coded so that the source code could not be easily copied by other parties. One of the ways to encode the source uses a combination of the Merkle-Hellman Knapsack algorithm, Discrete Logarithm and character modification to obtain complex encryption.

1. Introduction
In online transportation services as measured by physical quality, company quality, technical quality and interactive quality affect customer satisfaction [1]. With this in mind, in improving services in applications for customer satisfaction is the security of the application. The application is built on a web basis using the PHP programming language. The PHP programming language is currently widely used in several fields such as education, health, government and others. To increase security and user convenience there are several PHP frameworks such as Laravel, Codeignither and others. The PHP programming language produces web-based applications, where these applications can be accessed anywhere and anytime. So that security in the programming language needs to be considered because important information is contained in each web. Such as the vote counting application in the KPU called sirekap, which contains information about the election winner's votes, a hospital application that has confidential patient data and patient payments. In technology, no system is safe because every system has its own loopholes. What can be done is to minimize the leakage of information from the existing loopholes by performing cryptographic techniques. Cryptographic techniques can be performed on database storage and application source code. In this case, the encrypted source code is the PHP programming language. Source code is a collection of statements or declarations in computer programming language that are written and can be read by humans. The source code allows the programmer to communicate with the computer using several predefined commands. Cryptography is a field of science that studies how to secure data or information by encoding it or making it unreadable or re-understanding its meaning to be safe from attacks [2]. The encryption process is a process carried out to change the plaintext or initial characters into unrecognizable characters [3]. The encryption process is a process carried out to change the plaintext or initial characters into unrecognizable characters [4]. The Merkle-Hellman knapsack is an asymmetric algorithm, where there are public and
private keys. The way it works is to determine the key generation of the superincreasing line. The superincreasing sequence is a sequence where each value in the sequence is greater than the sum of all previous values [5]. Security of software source code has been tested using the Advanced Encryption Standard (AES) algorithm and the Base64 algorithm. The encryption process produces plaintext that contains the source code into unrecognizable characters. The decryption process also succeeds in returning unrecognizable characters to the initial plaintext containing the source code [6]. To maintain the confidentiality of digital messages, cryptographic techniques are used using XOR Chiper and Merkle-Hellman Knapsack. The tests were carried out successfully converting plaintext into ciphertext in the form of numbers. So that the attack on the XOR Chiper binary bit is safer against attacks [7]. Another study to secure source code used the Obfuscation technique. From this study, it can be seen that applications that use the obfuscation technique require a longer response time than those that do not use Obfuscation. [8]. Cryptographic techniques are effective enough to minimize attackers’ intentions to attack data because the data is encrypted. The encryption is converted into a digital image using the Merkle-Hellman Knapsack algorithm. Encoded data is prone to making the attacker suspicious, therefore it is better to combine it with steganography because it can be hidden in certain objects. [9]. In cryptographic e-commerce, a combined RC4 and Base64 algorithm is used to ensure the security of online transaction data carried out by customers because the customer's password at the bank has been disguised by an encryption process and is very difficult to crack if the key and calculation algorithm are different [10]. As security demands increase in terms of privacy and security, various technologies for withholding data are required. By combining one algorithm with another in order to get more complex results. When cryptographic and steganographic techniques are combined, communication will be safer [11]. The attack was carried out to test the Merkle-Hellman Knapsack algorithm which was different from Shamir's, the attack resulted in the latest algorithm in cryptographic techniques [12]. Seyma encryption proxies are used to analyze cryptanalytic attacks and check and validate desired security attributes such as confidentiality. The results prove that this scheme is susceptible and semantically insecure against IND-CCA or secured existentially against EUF-CMA in the specified security model [13]. Network security is a very important component in information system security because it is responsible for securing all information that passes through a networked computer [14]. ASCII Short for American Standard Code for Information Interchange, is a character encoding standard for communication devices. ASCII Table Codes represent text on computers, telecommunications equipment, and other devices.

2. Method

Algorithm combination has 2 public keys and 2 private keys, these keys are owned by the Merkle-Hellman Knapsack algorithm and Discrete Logarithm respectively, for the modified ascii table does not use a key but uses a table provided for encryption. The steps in the combination of the Merkle-Hellman Knapsack algorithm, Discrete Logarithm and modified Ascii Table are as follows:

- Specifies a superincreasing sequence where each element in the sequence must be greater than the number of the previous element. The sequence is used as the first private key.
- Choosing a prime number \( m \) on condition that it must be greater than the last element in the addition series.
- Choose the number \( n \), where the prime numbers are relative to \( m \).
- Generates the first public key with the following equation

\[
pi = si \times n \mod m
\]  

(1)

- Select the prime numbers p and q
- Calculating the value of \( n_{rsa} \) with the formula \( p \times q \)
- Calculating the value of \( \phi (n) \) with the formula \( (p-1) \times (q-1) \)
- Choose the number \( e \) with the condition \((e, \phi (n) = 1)\). The value \( e \) is used as the second public key
Calculating the value of d with the condition \( d \times e \mod (\phi(n)) = 1 \), the value of d is used as the second private key.

Perform the first encryption process with the following equation

\[
c = b_1*p_1 + b_2*p_2 + \ldots + b_n*p_n
\]  

(2)

Perform the second encryption process with the following equation.

\[
c = m \times mod \ n_{rsa}
\]  

(3)

The variable m in equation (3) is the result of the process of equation (2). From equation (2) and (3) it can be seen that the encryption process is carried out twice.

The results of equation 3 are continued by adjusting the ASCII table which has been modified and becomes the third encryption.

Table 1. The Ascii table is modified to produce a third encryption, the contents of the character fields can be adjusted as desired.

| Number | Character |
|--------|-----------|
| 0      | a         |
| 1      | b         |
| 2      | c         |
| 3      | D         |
| 4      | e         |
| 5      | F         |
| 6      | g         |
| 7      | h         |
| 8      | i         |
| 9      | j         |

The results of the third encryption in the modified ASCII table, the first decryption is done according to table 1.

Furthermore, a second decryption is performed using the following equation.

\[
e = m \times mod \ n_{rsa}
\]  

(4)

Perform the third decryption process using the following equation.

\[c \times n^{-1} \mod m\]

(5)

3. Implementation and testing

The garbage collection application is built using the PHP programming language, if it is run on the server side the source code must be uploaded so that the application can run. In order to increase security in the waste transportation application, security is required in the source code so that other parties have difficulty imitating the application.
Figure 1. The source code has been encrypted using a combination of algorithms to make it difficult to imitate the application.

```php
<?
  if (isset($_GET['send'])) {
    $string = 'Hello World';
    $result = base64_encode($string);
    // Further encryption logic...
  }
?>
```

Hello World

Figure 2. The application can be executed in the browser and produce according to the source code written.

To simplify calculations in the combination of these algorithms, values are assigned to existing variables according to applicable regulations.

**Table 2.** This table provides information about the values that have been given to existing variables.

| No | Information | Value |
|----|-------------|-------|
| 1  | Superincreasing | (1,2,4,8,16,32,64,128) |
| 2  | m           | 251   |
| 3  | n           | 11    |
| 4  | p           | 33    |
| 5  | q           | 41    |
| 6  | nrsa        | 1353  |
| 7  | φ (n)       | 1280  |

After the variable value is determined, it is followed by selecting the plaintext that will be encrypted and converted based on the existing Ascii table. This was done in order to start the first encryption process using the Merkle-Hellman Knapsack algorithm. Before the encryption process starts, use equation 1 to get the public key.
Table 3. This table provides information about the values that have been assigned to existing variables.

| Plaintext | Decimal | Biner      |
|-----------|---------|------------|
| e         | 101     | 01100101   |
| c         | 99      | 01100011   |
| h         | 104     | 01101000   |
| o         | 111     | 01101111   |
| "         | 34      | 00100010   |
| H         | 72      | 01001000   |
| e         | 101     | 01100101   |
| l         | 108     | 01101100   |
| l         | 108     | 01101100   |
| o         | 111     | 01101111   |
| 32        | 00100000 |
| W         | 87      | 01010111   |
| o         | 111     | 01101111   |
| r         | 114     | 01100101   |
| l         | 108     | 01101100   |
| d         | 100     | 01101100   |
| "         | 34      | 00100010   |
| ;         | 59      | 00111011   |

Table 4. This table shows the private key and public key. The private key is the value that exists in superincreasing. And the public key is obtained using the formula in equation 1: 

\[ pi = si \cdot n \mod m, \]

where \( pi \) is the resulting value in the form of a public key and \( si \) is the superincreasing or private key.

| Private Key | Public Key |
|-------------|------------|
| 1           | 11         |
| 2           | 22         |
| 3           | 44         |
| 4           | 88         |
| 16          | 176        |
| 32          | 101        |
| 64          | 202        |
| 128         | 153        |

Next do the calculations using equation 2 to get the first encryption.

Table 5. This table uses equation 2 below \( c = b_1 \cdot p_1 + b_2 \cdot p_2 + \ldots + b_n \cdot p_n \), where \( c \) is the first encryption, \( b \) is binary and \( p \) is the public key.

| Plaintext | Decimal | /Public Key |
|-----------|---------|-------------|
| e         | 101     | 0 1 1 1 0 0 1 0 1 320 |
| c         | 99      | 0 1 1 0 0 0 0 1 1 421 |
| h         | 104     | 0 1 1 0 1 0 0 0 0 242 |
| o         | 111     | 0 1 1 0 1 1 1 1 1 698 |
| "         | 34      | 0 0 1 0 0 0 1 0 246 |
| H         | 72      | 0 1 0 0 1 0 0 0 0 198 |
| e         | 101     | 0 1 1 0 0 1 0 1 320 |
| l         | 108     | 0 1 1 0 1 1 0 0 343 |
| l         | 108     | 0 1 1 0 1 1 0 0 343 |
| o         | 111     | 0 1 1 0 1 1 1 1 698 |
| 32        | 0 0 1 0 0 0 0 0 44 |
| W         | 87      | 0 1 0 1 0 1 1 1 566 |
| o         | 111     | 0 1 1 0 1 1 1 1 698 |
| r         | 114     | 0 1 1 0 1 0 1 0 356 |
| l         | 108     | 0 0 1 1 0 1 1 0 435 |
| d         | 100     | 0 1 1 0 1 0 0 0 167 |
| "         | 34      | 0 0 1 0 0 0 1 0 246 |
| ;         | 59      | 0 0 1 1 1 0 1 1 663 |
Then the calculation is done to get the second encryption using the modified Ascii table to get the third encryption.

**Table 6**. Then the calculation is done to get the second encryption using the modified Ascii table to get the third encryption.

| Plaintext | Encrypt 1 | Encrypt 2 | Encrypt 3 |
|-----------|-----------|-----------|-----------|
| e         | 320       | 1310      | bDba      |
| c         | 421       | 691       | gjb       |
| h         | 242       | 836       | iDg       |
| o         | 698       | 575       | FhF       |
| "         | 44        | 506       | Fag       |
| H         | 198       | 1254      | bcFe      |
| e         | 320       | 1310      | bDba      |
| l         | 343       | 403       | eaD       |
| l         | 343       | 403       | eaD       |
| o         | 698       | 575       | FhF       |
| "         | 246       | 984       | jie       |
| W         | 566       | 80        | ia        |
| o         | 698       | 575       | FhF       |
| r         | 356       | 302       | Dac       |
| l         | 435       | 403       | eaD       |
| d         | 167       | 260       | cga       |
| "         | 246       | 984       | jie       |
| ;          | 663       | 1329      | bDcj      |

After the encryption process is carried out 3 times, the decryption is also done 3 times, the first decryption is the conversion of the encryption value 3 using the modified Ascii table and the second decryption uses equation 4.

**Table 7**. This table uses table 1 for conversion of the third encryption to the first decryption, and use the following equation 4 $e = m \cdot c \mod n \cdot rsa$ to get the second decryption.

| Encrypt 3 | Decrypt 1 | Decrypt 2 |
|-----------|-----------|-----------|
| bDba      | 1310      | 320       |
| gjb       | 691       | 421       |
| iDg       | 836       | 242       |
| FhF       | 575       | 698       |
| Fag       | 506       | 44        |
| jie       | 984       | 246       |
| bcFe      | 1254      | 198       |
| bDba      | 1310      | 320       |
| eaD       | 403       | 343       |
| eaD       | 403       | 343       |
| FhF       | 575       | 698       |
| Fag       | 506       | 44        |
| ia        | 80        | 566       |
| FhF       | 575       | 698       |
| Dac       | 302       | 356       |
| eaD       | 403       | 435       |
| cga       | 260       | 167       |
| jie       | 984       | 246       |
Equation 5 is used to get the third decryption process, this process ends the decryption process from the algorithm combination.

Table 8. This table uses the following equation $5c \ast n-1 \mod m$ to get the third description.

| Encrypt 3 | Decrypt 1 | Decrypt 2 | Decrypt 3 |
|-----------|-----------|-----------|-----------|
| bDba      | 1310      | 320       | 166       |
| gb        | 691       | 421       | 198       |
| iDg       | 836       | 242       | 22        |
| FhF       | 575       | 698       | 246       |
| Fag       | 506       | 44        | 4         |
| jie       | 984       | 246       | 68        |
| bcFe      | 1254      | 198       | 18        |
| bDba      | 1310      | 320       | 166       |
| eaD       | 403       | 343       | 54        |
| eaD       | 403       | 343       | 54        |
| FhF       | 575       | 698       | 246       |
| Fag       | 506       | 44        | 4         |
| ia        | 80        | 566       | 234       |
| FhF       | 575       | 698       | 246       |
| Dac       | 302       | 356       | 78        |
| eaD       | 403       | 435       | 108       |
| cga       | 260       | 167       | 38        |
| jie       | 984       | 246       | 68        |
| bDcj      | 1329      | 663       | 220       |

After getting the value from the third decryption, the next step is to construct the binary value based on the largest private key.

Table 9. This table describes the function of the Merkle-Hellman private key to retrieve the decimal value of characters based on the ASCII table. Arrange binary from right to left.

| Decrypt 3 | / Private Key | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 | Biner | Decimal |
|-----------|---------------|-----|----|----|----|---|---|---|---|------|---------|
| 166       | 1 0 1 0 0 0 1 1 0 01100101 | 101 |
| 198       | 1 1 0 0 0 1 1 0 01100111 | 99  |
| 22        | 0 0 0 0 1 0 1 1 0 01101000 | 104 |
| 246       | 1 1 1 1 0 1 1 0 1 01101111 | 111 |
| 4         | 0 0 0 0 0 0 1 0 0 00100000 | 32  |
| 68        | 0 1 0 0 0 0 1 0 0 00100010 | 34  |
| 18        | 0 0 0 0 1 0 0 0 0 01001000 | 72  |
| 166       | 1 0 1 0 0 0 1 1 0 01100101 | 101 |
| 54        | 0 0 1 1 0 1 1 1 1 01101100 | 108 |
| 54        | 0 0 1 1 0 1 1 1 1 0 01101100 | 108 |
| 246       | 0 1 1 0 0 1 1 1 1 01101111 | 111 |
| 4         | 0 0 0 0 0 0 1 0 0 00100000 | 32  |
| 234       | 1 1 1 1 0 1 0 1 1 01010111 | 87  |
| 246       | 1 1 1 1 0 0 0 1 1 01101111 | 111 |
| 78        | 0 1 0 0 1 1 1 1 1 01110010 | 114 |
After getting binary values in Table 9, it will obtain the initial plaintext related to the ASCII table.

### Table 10: This table is the result of the decryption process with the initial plaintext "echo "Hello World";"

| Biner | Decimal | Plaintext |
|-------|---------|-----------|
| 01100101 | 101 | e |
| 01100011 | 99 | c |
| 01101000 | 104 | h |
| 01101111 | 111 | o |
| 00100000 | 32 | |
| 00100010 | 34 | " |
| 01001000 | 72 | H |
| 01100101 | 101 | e |
| 01101100 | 108 | l |
| 01101100 | 108 | l |
| 01101111 | 111 | o |
| 00100000 | 32 | |
| 01010111 | 87 | W |
| 01101111 | 111 | o |
| 01110010 | 114 | r |
| 01101100 | 108 | l |
| 01101100 | 100 | d |
| 00100010 | 34 | " |
| 00111011 | 59 | ; |

The following details can be seen from the encryption process for each algorithm.

### Table 11: Result of each algorithm steps

| No | Algorithm | Result |
|----|-----------|--------|
| 1  | Plaintext | echo "Hello World"; |
| 2  | Merkle-Hellman | 3204212426984424619832034334369845666698356343 |
| 3  | Discrete | 13106918365750698412541310434035755068057530 |
| 4  | Algorithm | bDbaAgibAlDgAFhFAFagAjieAbcFeAbDbaAeaDAeaD |
|    | Combination | AFhFAFagAiaAFhFADacAeaDAcgaAjieAbDcj |

### 4. Conclusion

Combination of algorithms (CEST Cryptography) can be applied to the source code of an application to increase the security of the application. The source code is protected because some important data has been encrypted with 2 private keys and 2 public keys so that it is not easily read by unauthorized people so that the source code is safer and not easily copied. Implementing a combination of algorithms on a system requires access time longer than not. Because when using a combination of algorithms, it requires 3 times the encryption process and 3 times the decryption process, if you want
to apply it to a system, it is expected that the source code is very important so that the access speed is not too long.

5. References

[1] A. A. Purwati and M. L. Hamzah, “Total service quality management and it’s impact on customer satisfaction and loyalty of online transportation in Indonesia,” Int. J. Sci. Technol. Res., 2019.

[2] G. Singh and S. Supriya, “A Study of Encryption Algorithms (RSA, DES, 3DES and AES) for Information Security,” Int. J. Comput. Appl., 2013, doi: 10.5120/11507-7224.

[3] F. N. Pabokory, I. F. Astuti, and A. H. Kridalaksana, “Implementasi Kriptografi Pengamanan Data Pada Pesan Teks, Isi File Dokumen, Dan File Dokumen Menggunakan Algoritma Advanced Encryption Standard,” Inform. Mulawarman J. Ilm. Ilmu Komput., 2016, doi: 10.30872/jim.v10i1.23.

[4] A. R. Tulloh, Y. Permanasari, and E. Harahap, “Kriptografi Advanced Encryption Standard (AES) Untuk Penyandian File Dokumen,” J. Mat. UNISBA, 2016.

[5] A. Aminudin, A. F. Helmi, and S. Arifianto, “Analisa Kombinasi Algoritma Merkle-Hellman Knapsack dan Logaritma Diskrit pada Aplikasi Chat,” J. Teknol. Inf. dan Ilmu Komput., 2018, doi: 10.25126/jtiik.201853844.

[6] C. Saefudin, G. Abdillah, and A. Maspupah, “PENGAMANAN SOURCE CODE PROGRAM MENGGUNAKAN ALGORITMA ADVANCED ENCRYPTION STANDARD DAN ALGORITMA BASE64,” Semin. Nas. Apl. Teknol. Inf., 2019.

[7] O. K. Sulaiman, “HYBRID CRYPTOSYSTEM MENGGUNAKAN XOR CIPHER DAN MERKLE-HELLMAN KNAPSACK UNTUK MENJAGA KERAHASIAAN PESAN DIGITAL,” J. Teknol. Inf., 2019, doi: 10.36294/jurti.v3i2.1072.

[8] M. Maskur, Z. Sari, and A. Miakhir, “Implementation of Obfuscation Technique on PHP Source Code,” Proceding Electr. Eng. Comput. Sci. Informatics, 2018, doi: 10.11591/eccisi.v5i1.1729.

[9] A. Lestari, A. S. Sembiring, and T. Zebua, “TEKNIK PENYEMBUNYIAN PESAN TEKS TERENKRPSI ALGORITMA MERKLE-HELLMAN KNAPSACK MENGGUNAKAN METODE PIXEL VALUE DIFFERENCING KE DALAM CITRA DIGITAL,” KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer), 2019, doi: 10.30865/komik.v3i1.1590.

[10] F. W. C, A. P. Rahagiar, and F. Fretes, “Penerapan Algoritma Gabungan Re4 Dan Base64 Pada Sistem Keamanan E-Commerce,” Semin. Nas. Apl. Teknol. Inf., 2012, doi: 10.1007/s10649-005-5536-8.

[11] J. K. and R. Khan, “Review on Network Security and Cryptography,” Int. J. Adv. Res. Comput. Sci. Softw. Eng., 2018, doi: 10.23956/ijarcss.v8i6.712.

[12] J. Liu, J. Bi, and S. Xu, “An Improved Attack on the Basic Merkle-Hellman Knapsack Cryptosystem,” IEEE Access, 2019, doi: 10.1109/ACCESS.2019.2913678.

[13] A. Waheed et al., “Cryptanalysis and Improvement of a Proxy Signcryption Scheme in the Standard Computational Model,” IEEE Access, 2020, doi: 10.1109/ACCESS.2020.3009351.

[14] M. R. Joshi and R. A. Karkade, “Network Security with Cryptography,” Int. J. Comput. Sci. Mob. Comput., 2015.

Acknowledgments

The author is grateful to Allah SWT who has provided fluency in both writing and ideas that arise and would like to thank the Indonesian Putra university YPTK Padang who funded research and colleagues who helped in completing the research so that the author could provide the best results by finding a combination of algorithms for source code security. It is hoped that further development can be better in its application.