Advances in Science, Technology and Engineering Systems Journal Vol. 3, No. 5, 311-320 (2018)

www.astesj.com

Special Issue on Multidisciplinary Sciences and Engineering

ASTESJ

ISSN: 2415-6698

Protecting Private Data using Improved Honey Encryption and Honeywords Generation Algorithm

Thanda Win*, Khin Su Myat Moe

Yangon Technological University, Department of Computer Engineering and Information Technology, 11010, Myanmar

ARTICLE INFO

Article history:
Received: 14 August, 2018
Accepted: 02 October, 2018
Online: 14 October, 2018

Keywords:
Private data
Honey encryption
Honeywords
Distribution Transforming Encoder

ABSTRACT

Nowadays, many companies and organizations use various encryption techniques to secure their private data before send it through the unsecure network. However, many attackers try to store private data by using various attacks. Most of the organizations use password based encryption algorithm (PBE) to protect their private data. However, the existing PBE methods are vulnerable in brute-force attacks because of user-generated weak or repeated passwords. The problem of weak password based encryption algorithm prompted us to introduce honey encryption (HE). Honey encryption and honeywords help to minimize this vulnerability of password based encryption algorithm and they can prevent the text based messages from brute face attacks in order to make messaging communication more secure and efficient. However, the conventional HE has message space limitation in Distribution Transforming Encoder (DTE) process and storage overhead problem in honeywords generation process. Therefore, our proposed honey encryption algorithm uses discrete distribution function in DTE process to solve message space limitation instead of using cumulative distribution function. Storage overhead and typo safety problem can overcome by using our proposed honeywords generation method in honey encryption algorithm. Furthermore, we also propose hashing and salting algorithm for securing password and key expansion. In this paper, we describe the case studies as a calculation of DTE and hashing and salting algorithm. And then, we design and implement the improved honey encryption mechanisms and honeywords generation algorithm in message transmission process. Finally, we show the comparison results using mathematical model in DTE process and security analysis of our system.

1. Introduction

Most of organizations over the world want to exchange their private messages secretly over the unsecure communication channel. So, they use end to end encryption services with various data encryption algorithm for their private data. One type of encryption process called the end-to-end encryption (E2EE) can read the messages which own the secrete keys [1]. Most of the organizations use password based encryption (PBE) algorithm because user can select and remember their key easily. However, the existing PBE algorithms are weak because hackers can get the message easily if they find the keys using the various attacks. Among them, brute force attack is the vulnerable to the applications and websites using encryption algorithm because these attacks attack the password files in the database until the real key or passwords find [2]. So, many researchers try to prevent the brute force attacks by using various methods.

The two countermeasure methods to brute force attacks have been applied by many modern systems. The first method is to enhance the time complexity for attackers. When the time complexity increases, the attackers take many computation times for cracking the passwords and many systems protect the data by increasing key length pseudorandom number generator (PRNG). Therefore, many hashing algorithm such as MD5 and SHA256 are invented for improving computations times [3]. However, the brute force attack can reveal the key within the polynomial time. So, the honeywords generation methods are used in many cryptosystem for protecting the keys. The attackers can face the difficulties and can't stole the password file if the system uses honeyword generation method that stores the honeywords with the real passwords in the password files. The purpose of decoy passwords or honeywords are used for the attackers that attack the hashing password file using various attacks especially brute force attacks. So, the existing honeywords generation methods generate

*Thanda Win, Yangon , +959797102699, thanda80@gmail.com

at least 20 honeywords for one real password [4]. So, the existing method meets the storage overhead problem.

The second method is statistical coding scheme that can produce false plaintext messages [3] using ASCII code table. But, the attacker can easily know this false message is not the real message because these messages are meaningless messages. HE is an encryption algorithm that can produce a ciphertext which when decrypted by any wrong key by the brute force attacker, then it looks like meaningful fake plaintext which is not the original message. So this encryption provides security against brute force attack. Distribution transforming encoder (DTE) is the main part of HE and it performs the mapping of the message space into the seed space of binary bits string. According to the probabilities of a message in the message space, it maps the message to a seed range using DTE process in a seed space randomly. Then the resulting seed space is XORed with the key to get the ciphertext. For decryption, the ciphertext is XORed with the key and the seed is obtained. Then DTE uses the seed location to map it back to the original plaintext message. Even if the key is incorrect, the decryption process produces a honey message from the message space and thus can deceive the attackers [4]. In this existing HE algorithm, the message space can't accept more than four messages.

This paper described three contributions. Firstly, we report new DTE process for overcoming message space limitation problem in HE algorithm. And then, we design and implement the improved honey encryption system using new DTE process and apply the concept to message transmission application. These applications are based on uniformly distributed message spaces and the symmetric encryption mechanism. We also propose the new honeywords generation algorithm and hashing algorithm. Secondly, we apply two countermeasure attacks such as honey encryption and honeywords generation algorithms in message transmission process. Thirdly, we discuss comparison results of our proposed system and existing systems.

2. Related Works

For protecting the brute force attack, we firstly described two different methods such as hashing algorithm and statical code scheme using ASCII code table. However, these two methods can't fully against the brute force attacks. So the honeywords generation method and honey encryption algorithm are reported in the introduction. Although two methods can protect the brute force attack, they have weakness in their process. In this part, we describe the weakness of password hashing and ASCII code table. And then, we described the limitation problem of honeywords generation methods and honey encryption process as follows.

In the past year, D. Gross described the dangerous hashing passwords that are used in the various websites such as Evernote, Yahoo, LinkedIn and many other websites [5]. Due to the password cracking technique proposed by M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek, the hashing password can easily know by the attackers [6]. P.G. Kelley, S. Komanduri, M.L. Mazurek, R. Shay, T. Vidas, L. Bauer, N. Christin, L.F. Cranor, and J. Lopez reported most of the passwords of users are weak and easy guessed password. So, their proposed method can easily know the hashing passwords of user [7]. The attackers can get three billion guesses per second if the system using some hashing algorithm [8]. Therefore, passwords are stored using hashing algorithm in the database are not against the various attacks.

S. Kharod, N. Sharma and A. Sharma proposed a hashing method using salting and differential masking that can provide the better security of the password with the faster time [9]. Although this method is better method for the passwords with better processing time, it can't strongly protect the brute force attack.

Above the description, hashing algorithm that uses for increasing computation time and producing false plaintext messages are not fully protect the various attacks especially brute force attack. Therefore, honeywords generation algorithm appeared in 2013 and honey encryption described in 2014.

In order to improve the safety of the hashed passwords, A. Juels and R. L. Rivest suggested a new method called honeywords generation method. They maintain the honeywords with the real password for each user's account by creating honeywords. The attacker cannot classify which password is real password if he gets inversion file of hashed password or honeywords. The auxiliary server or honey checker can classify the real password and honeyword for login process and will set off an alarm to categorize between the honeywords and real password [10]. However, this method can cause typing mistake of users during entering of password because honeywords generation method create the similar honeyword with the real password. The main weakness of this method is storage overhead problem.

Due to the development of graphical processing unit (GPU), the adversary can solve the hashing password files. So, Z. A. Genc, S. Kardas and M. S. Kiraz proposed the new method by keeping the decoy password and real password into the database. In this work, the author described the honeychecker for testing entering the false passwords to the system. Although getting the password file and coverting the hash code into the password, he can't enter the system without passing the honeychecker [11]. Therefore, this method can against the attacks of attackers, it meets the storage overhead problem.

For making less the existing problem of honeywords generation method, N. Chakraborty and S. Mondal was created the honey circular list algorithm. In this paper, the authors described circular list for storage of hashing password. When the attacker gets the password file, he cannot compute the distance of this password and he cannot login into the system. The honey circular list method can make less the storage problem of earlier existing algorithms [12]. However, this method can't almost solve the problem of honeywords producing methods.

S. R. Shinge and R. Patil presented a data encryption and decryption that uses values in ASCII code table. In this ASCII code table encryption approach, the plaintext messages and encryption keys are converting into the string using ASCII values [13] for producing ciphertexts. If an adversary stoles the key file and then he tries to decrypt the ciphertext use one key in this key files, he gets the meaningless ciphertext if he uses the false key. So, the attacker can know easily he uses the false key in this process.

A new encryption technique that can protect brute force attack called honey encryption (HE) algorithm was proposed by N. Tyagi, J. Wang, K. Wen and D. Zuo. When the attacker tries to get decrypted data by using brute force attack, that technique

can deceive the attacker and can give bogus meaningful messages. A new encoding and decoding scheme called a distribution transforming encoder (DTE) are used by HE and it has the limitation for assigning the plaintext messages [4]. The DTE in that system has limitation for placing plaintext message into the seed space.

According to the previous works, the existing system meets the storage overhead and message space limitation problems. Therefore, we propose a system to overcome message space limitation and storage overhead problem. This honeywords generation system can make less storage cost compared to the existing honeywords generation algorithm. Moreover, it can easily overcome the typo safety problem. For securing password files, we use new hashing and salting algorithm. The hashing and salting time is faster than the existing MD5 hashing algorithm and hashing algorithm using salting and differential masking . Finally our system using new honeywords generation and hashing algorithm, new DTE process can overcome message space limitation problem using DTE compared with the existing honey encryption algorithm.

3. Proposed Methodology

This part reports about the methodology of the proposed techniques such as the user authentication and login process, honeywords generation method, hashing and salting algorithm, distribution transforming encoder and honey encryption algorithm.

3.1. User Authentication and Login Process

In any login procedure, when the user is a new member, he needs to make the registration process. Otherwise, the user enters the system with his username and password. After making the registration process, the system produces the honeywords for this member using honeywords generation algorithm and the sweetwords that includes honeywords and real password are kept into the password file using our proposed hashing and salting algorithm. The registration process and login process are shown in the Figure 1(a) and 1(b). After the registration process, he can login into the system with his password.

The server detects the login password whether it has in the database or not. If the password exists in the database, the password is sent to the honeychecker by the server for categorization of honeyword and actual password. When the login password doesn't equal to the user's real password that stores in the database, administrator can get the alarm message from the honeychecker. Otherwise, the user can enter into the system successfully.

3.2. HoneyChecker

Honeychecker is an auxiliary secure server that can be used for categorisation of sweetwords including real passwords and false passwords. The honeychecker stores secret information such as the index of real passwords and the hashing passwords. The main server can communicate with the honeychecker when a user enters the system. When the false password or honeyword is entered into the system, the system administration can know immediately by raising an alarm signal from the honeychecker. In our system, the honeychecker performs two main processes. The first process is to classify the honeywords or real password when the user logging the system. The second is to send an alarm message to the administrator if the attacker enters with the honeywords.

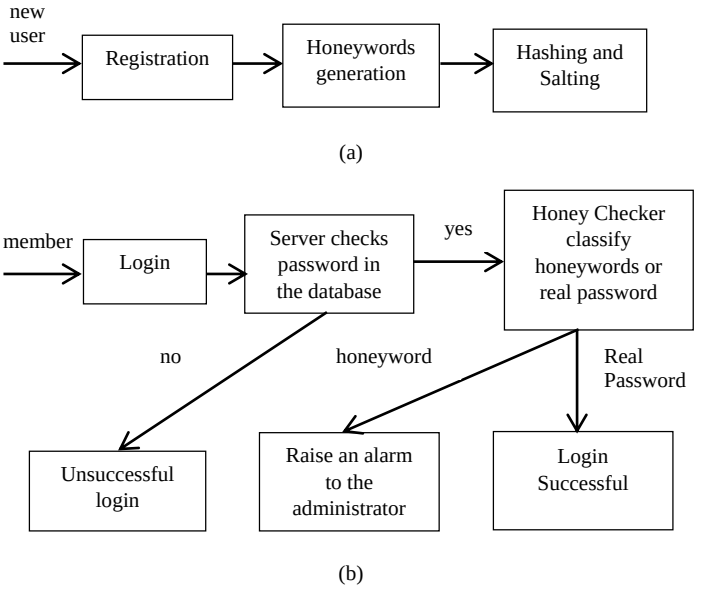

Figure 1: (a): User Registration Process (b): User Login process

3.3. Honeywords Generation Method

For every web application, passwords are notoriously weak authentication in the authentication process because users choose easily guess passwords by the attackers that apply dictionary, rainbow table, brute force attacks etc.. Therefore, as many technologies such as graphic processing unit (GPU) are developing, the attackers can get the user's password and can crack easily hashing password without consuming many times. So, the real passwords and the honeywords that are generated from our proposed honeywords generation algorithm are kept using our proposed hashing and salting algorithm for each account in the database by the system. The purpose of honeywords generation methods generates the honeywords to deceive the attackers [14]. However, the existing honeywords generation algorithms have the storage overhead problem.

Table 1. Example of password file P1 in main server

user name	honeyword index set
David	(21,35,40,55)
Suzy	(18,35,40,55)
Jane	(18,21,40,55)
Joy	(18,21,35,55)
August	(18,21,35,40)

Table 2. Example of password file P2 in honey checker

real password index	hashing password
18	763f7b09c8ac104bd
21	3e8fdfcd7f3615428b0c
35	7e8fcb6b2bc
40	365b4f15a8d445d
55	984d078ac47c2c82987d19

So, we proposed the new honeywords generation algorithm for making less the storage cost. In our proposed algorithm, we denote the member's passwords of our system as honeyword without creating and storing the individual honeywords in the database. Therefore, we create two password files P1 and P2 in the server's database and honeychecker's database. The password file P1 that includes username and honeyword index set in the main server's database. Real password index and hashing passwords including in password file P2 are kept by the system in the honeychecker's database. The table is sorted randomly according to the user registration position. The example storing of honeywords and real passwords are shown in table 1 and table 2.

The detail procedure of honeywords generation is described as follows.

Inputs:

1. username u_i and password p_i

2. After creating accounts, index list of real password and honeywords need to assign

Procedure

Step 1: User account registration and honeywords creation

a. P1 and P2 password files creation

P1 stores username and honeyindex set h_i, P2 keeps the index of real password r_i and the hash of the corresponding user's passwords $H(p_i)$

b. The list index set of honeywords h_i and real password p_i are created like the following

P1= {<u_i,h_i>, <u_2,h_2>, ----------,<u_i,h_i>}

P2= {<r_1,$H(p_1)$>, <r_2,$H(p_2)$>,-------, <r_i,$H(p_i)$>}

Step 2: User passwords and fake password storage:

a. P1 including username and honeyindex are set in main server
b. P2 including index of real password and hashing passwords are set in the honey checker

Step 3: HoneyChecker Classification

Set r_i, $H(p_i)$

Sets correct password index r_i for the user u_i

Set: Login password k

Check: r_i, k

If r_i is equal to the login password k, login process successful.

Otherwise, the honeychecker sends an alarm message to the administrator.

3.4. Hashing Algorithm

In our proposed system, honeywords and real passwords are stored into the database using our proposed hashing algorithm for providing better security of key or password with faster time. The detail steps of our proposed hashing algorithm are described the following.

Step 1: Change the user's entering password string into binary bits.

Step 2: Binary bits string are appended with padding bits 00000000.

Step 3: Change the string of binary bit 1 into 0 and 0 into 1.

Step 4: The changing string are make the XOR operation with the binary string by concatenation of 0's and 1's.

Step 5: The string rotates from right by 4 characters.

Step 6: The resulting binary string converts to hexadecimal string and stores into the password file that locates in the database.

The example of our hashing algorithm is shown as following. In this example user chooses her password as December30 and our proposed hashing algorithm converts into the hexadecimal string using the following steps.

Step1: Convert the input expression (password: December30) into binary string.

01000100 01100101 01100011 01100101 01101101 01100010 01100101 01110010 00110011 00110000

Step 2: Before making the password hashing, binary bits string are appended with padding bits 00000000 (green color).

00000000 01000100 01100101 01100011 01100101 01101101 01100010 01100101 01110010 00110011 00110000

Step 3: Change the string of binary bit 1 into 0 and 0 into 1.

11111111 10111011 10011010 10011100 10011010 10010010 10011101 10011010 10001101 11001100 11001111

Step 4: The changing string are performed the XOR operation with the binary string by concatenation of 0's and 1's.

String1:

11111111 10111011 10011010 10011100 10011010 10010010 10011101 10011010 10001101 11001100 11001111

String 2:

00001111 00001111 00001111 00001111 00001111 00001111 00001111 00001111 00001111 00001111

After XORing operation, the resulting binary string are described as follows.

11110000 10110100 10010101 10010011 10010101 10011101 10010010 10010101 10000010 11000011 11000000

Step 5: The resulting string rotates from right by 4 characters. So, first 4 characters are rotated to the right.

00001111 00001011 01001001 01011001 00111001 01011001 11011001 00101001 01011000 00101100 00111100

Step 6: The resulting rotate string converts to hexadecimal string and stores into the password file that locates in the database. So, the resulting hexadecimal string is as

follows.

Hexadecimal String - 7B49593959D929582C3C

3.5. Distribution Transforming Encoder

Distribution transforming encoder (DTE) process is the main key of honey encryption algorithm that assigns the message space M to the binary bits string of seed space S. Depending on the number of messages, the range of seed space assigned for distribution. The existing DTE process uses cumulative distribution function to map the message space into the seed space and it meets the message space limitation problem [4]. The DTE process includes encoding and decoding. In the encoding process, the DTE apply the discrete distribution function for assigning the plaintext messages M to the seed space S. The DTE process encodes the plaintext messages M to the range of seed space S is assigned randomly. On the other hand, the decoding process is harder than encoding process. In the decoding process, the seed space S decodes into the message space M using DTE process. One of our main contributions is to construct the DTE process using discrete distribution function to overcome message space limitation problem.

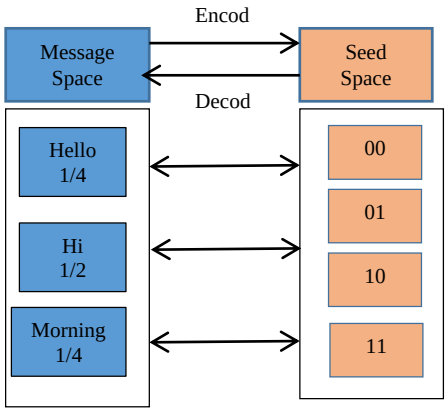

Figure 2: Basic Example of DTE Process

Figure 2 illustrates a basic example of DTE process. In this figure, message space consists of different elements in this case M={ hi, hello, morning}. Depending on the number of messages, the probability of each element using discrete distribution function is generated.

3.6. Honey Encryption Algorithm

Password-based encryption (PBE) and hashing techniques cannot fully against the brute force attacks since most of the users that use password as encryption key select the weak and repeatedly user-generated passwords. Honey encryption (HE) algorithm is one of the deceive algorithm to attackers who use brute force attack. The purpose of HE algorithm is to lure the attackers. HE generates the honey message through the brute force attack and deceives the attackers in a way that they cannot classify which messages are true messages. The core creation of existing HE is the distribution transforming encoder (DTE) that uses cumulative distribution function. Constructing the better DTE process can provide to get better HE algorithm [4].

The contribution of this paper consists of three main categories: honeywords generation, improved hashing algorithm and distribution transforming encoder. In our honeywords generation algorithm, storage overhead problem can be reduced. Our improved hashing algorithm is faster than existing MD5 hashing algorithm. Our improved DTE process that uses discrete distribution function can improve the message space limitation. Figure 3(a) and 3(b) show the basic encryption and decryption process of honey encryption algorithm.

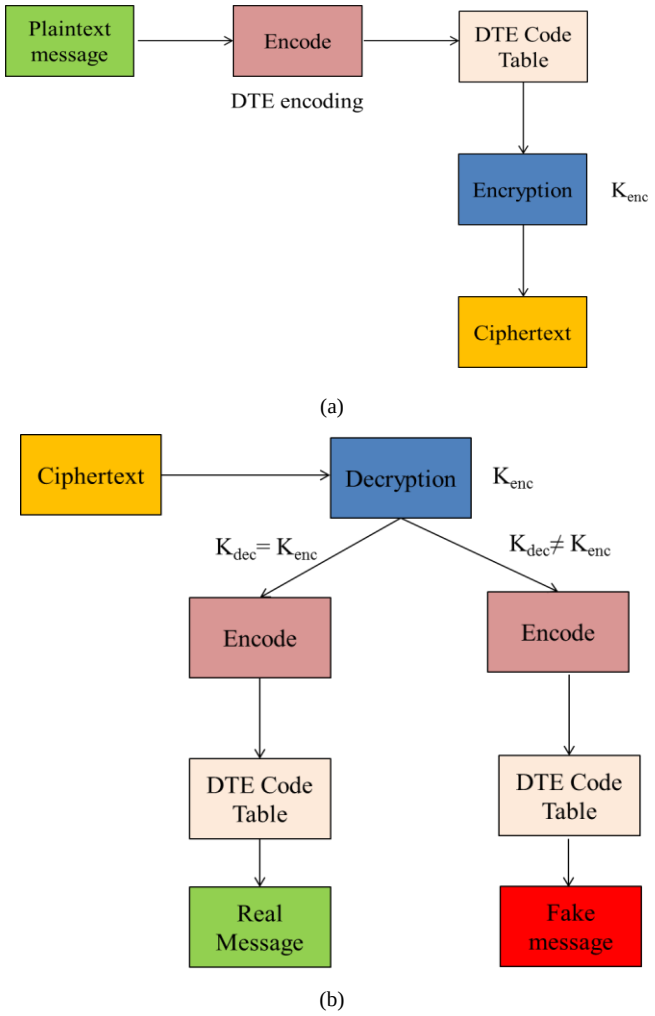

Figure 3 (a) Basic Encryption Process of HE Algorithm (b) Basic Decryption Process of HE Algorithm

The encryption process includes the plaintext message M encoding process using encoding code table and encrypting with encryption keys K_{Enc} to produce the ciphertext. The code table is made by using DTE process that use discrete distribution function. The sender sends resulting ciphertext to the receiver via the internet.

In the decryption process, the receiver decrypts the ciphertext with K_{Dec}. The decryption result decodes with the code table that equals the code table using in encryption process. If encryption key K_{Enc} and decryption key K_{Dec} are the same, the receiver can gets a *real* message that the sender sends. Otherwise, the receiver will get a bogus message that look like real message.

4. Proposed System Implementation

This section discussed about the flowchart of our proposed system, system design, example of the proposed system and mathematical model of the proposed system.

4.1. Flowchart of Proposed System

Figure 4 describes the detail flowchart of the proposed system.

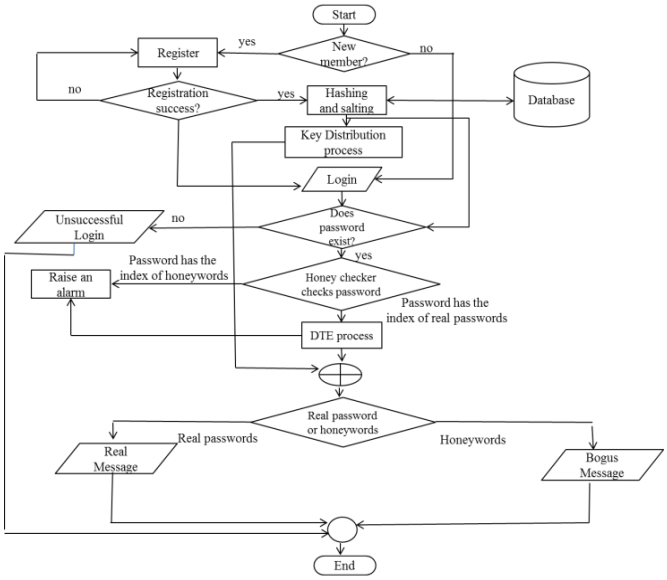

Figure 4: Flowchart of Proposed System

If a new user wants to enter this system, he needs to make the registration process at least one time. He doesn't need to make the registration process to enter the system if he is a member of the system. After making the registration process, he can login into the system with his username and password. If he enters into the system with his username and password, the system checks his password exists or not in the database. In this stage, he can get three conditions. Firstly, he can get unsuccessful login message from the system if his password doesn't have in the database. At that time, he needs to check his username and password if he is a member of this system. Otherwise, he needs to make registration process. His password will be sent by the server sends his password to honey checker for categorization of honeyword or sugarword if his password exists in the database. Secondly, if his password is real password, he will perform the DTE process and gets the real message from the sender's send. Otherwise, he will perform the DTE process and he will get the bogus message.

4.2. System Design of the Proposed System

The process of our proposed system includes DTE process and key or password distribution process. DTE process makes encoding and decoding of original plaintext message. DTE that uses discrete distribution function for encoding message space M also called plaintext message such as $5000, $10000, $750, $900, $200 into the seed space S. The range of seed space denotes as binary bits. The number of plaintext message space depends on the directly proportional to the range of seed space. For placing the number of messages m in plaintext message M to seed space S_m, DTE requires discrete distribution function in order to overcome message limitation.

In keys or passwords distribution, honeywords generation and key expansion of this password file of honeywords and real passwords are considered. In the honeywords generation process, we generate the honeywords by denoting the other user's real passwords and stores as honey indexes for overcoming weakness of existing honeywords generation methods such as storage overhead problem and typo safety problem. By using the other's sweetword as decoy password, the adversary becomes complicated which one is correct password or honeywords. In this case, the password and honeywords are stored in the password file using our proposed hashing algorithm to provide better security. And then, the resulting password and honeywords are randomly placed into the seed space S_k using DTE process. After placing password file into the seed space S_k, the key expansion result S_k is XORed with the message distribution result S_m of DTE. Finally, the ciphertext message is successfully created. During the decryption process, the ciphertext is XORed with the secret key to get the seed space S_m. And then we decode this seed space S_m to get original plaintext message.

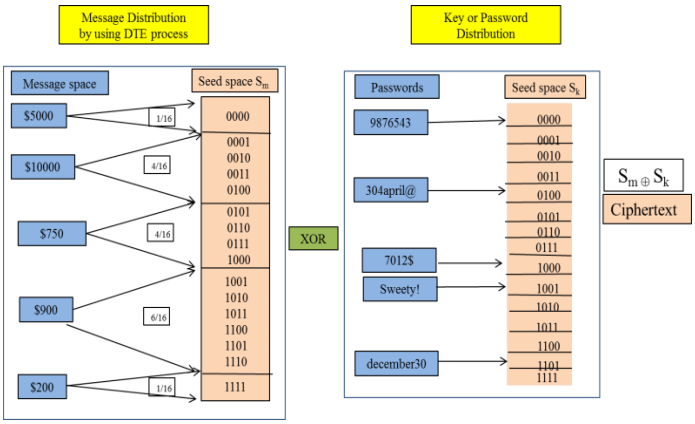

Figure 5: Proposed System Design

4.3. Example of Our Proposed System

Figure 5 describes the detail example design process of our proposed system. In this example, different messages such as M = {$5000, $10000, $750, $900, $200} are denoted as message space. In this case, message space M has five elements. According to the number of messages, probabilities are assigned to each element. By calculating, the seed space range of message M is $2^{5-1} = 2^4 = 16$ and seed space is 4-bits binary strings. Depending on the results of probabilities values, different messages elements are mapped into the range of seed space by the system. For assigning the message space, the messages are ordered randomly. With this randomly ordering, the messages get different probabilities and the range of seed space will be different. The secrete key or passwords are K = {9876543, 30april@, 7012$, Sweety!, december30}.

Using discrete distribution function, probability of the messages is {1/16, 4/16, 6/16, 4/16, 1/16}. So, the seed space S_m of the message "$5000" is {0000} and "$10000 is {0001, 0010, 0011, 0100}. And then, seed space S_m of $750 is {0101, 0110, 0111, 1000, 1001, 1010} and $900 is {1011, 1100, 1101, 1110}. Finally, the seed space S_m of $200 is {1111}. According key distribution process, the seed space S_k of 9876543 is {0000} and

april30@ is {0100}. The next 7012$ is {1000} and Sweety! is {1001}. And then, the seed space S_k of december30 is {1101}.

In the encrypting process, Bob wants to send encrypting message "$5000" to Alice under the secrete key as april30@. So, we compute seed space S_m by using DTE process and then the result S_m of $5000 is (0000).Similarly, according to the key or password distribution, seed space S_k of password april30@ is (0100). And then, Bob sends the resulting ciphertext (0100) by XORing the seed space S_m and S_k to Alice.

The decryption process is harder than encryption process because it includes message recovery process. In this process, Alice tries to decrypt ciphertext C by using password april30@ of Bob. Firstly, Alice computes seed space of key distribution S_k and this S_k (0100) is XORed with the ciphertext (0100) to get seed space of message distribution S_m (0000). And then, he decodes S_m by using DTE process to get original plaintext message "$5000".

On the other hand, attacker John wants to get Bob message. So he attacks password file by using brute force attack. John gets password file but he doesn't classify which is Bob password. So, he randomly chooses password 9876543 and tries to decrypt Bob message. John computes seed space of 9876543 to get key distribution S_k (0000). And then he makes XOR process of ciphertext C and seed space S_k. The resulting seed space of message distribution S_m is (0100). And then, he decodes S_m by using DTE process to get Bob message. Finally, John gets "$10000" after decoding process of seed space of message distribution. In this process, he doesn't know which message is true message or decoy message. This encryption algorithm can make confusion for attackers and it can strongly protect brute force attack.

4.4. Design Methodology

The methodology proposed in this research work is based on Honey encryption algorithm using honeyword Generation Method. The above mentioned work is divided into various modules as follows:

- Registration: At the time of registration, entering password, the system converts password into hash values and store into database. Along with hash values the real password's hash is also store at specific random position. After entering encryption key for encryption and decryption, system will generate honeywords for deceiving the attackers.
- Honeyword Generation: Honeywords are generated from the real password and in case any hacker tries to hack into the account by guessing the password the main user is sent alerts in form of a mail or some message so he knows that somebody is trying to login using his or her account.
- Login: If a user wants to login into the system, the user will type username and password. If the password matched with the hash password, the user is authenticated.
- Message Encryption and Decryption: User can send encryption message via the communication channel using encryption and decryption key.
- Generating honey message or fake message: If the encryption key is not match to the decryption key, the system will generate the honey message or fake message for counterattacking the attacker.
- Admin Management: Admin has the privileges to control over the mechanism. Admin has the authorization to maintain the user accounts and messages management.
- Hacker: If the hacker tries to break the system and enters any honeyword then the alert is given to the Actual user. When the attacker gets the encryption key file and he tries to decrypt the ciphertext using the honeywords, he will get the honey messages.

4.5. Mathematical Model of Our Proposed System

The core process of HE uses DTE for assigning the message space M into the seed space S_m using discrete distribution function in our proposed system. The discrete distribution function is a branch of probability function. The values z in the sample space Ω of discrete probability f(z) theory must have between 0 and 1 and the total values of f(z) for all values z in the sample space Ω must equal to one. This discrete probability theory satisfies the following two properties [15].

1. $f(z) \varepsilon [0,1]$ for all $x \varepsilon \Omega$
2. $\sum f(z) = 1$

- DTE using Discrete Distribution Function

We denote S be seed space or sample space and the value of seed space is [0,1]. According to the discrete distribution function, we calculate to get one for the total value of probability P(z) for all message in seed space S. Let n be total number of messages, x be number of '0" observed, M be the message space that contains number of messages and m be message. The following examples prove that DTE using discrete distribution function satisfies the properties of discrete probability theory. In this example we use number of five messages M={m1,m2,m3,m4,m5}. Firstly, we calculate seed space S.

Number of messages n= 5

For Distribution transforming encoder,

Seed space $S_m = 2^{n-1} = 2^{5-1} = 2^{5-1} = 2^4 = 16$

S={0000,0001,0010,0011,0100,0101,0110,0111, 1000,1001, 1010, 1011, 1100, 1101,1110,1111}

By using Discrete Distribution Function,

m_1=1/16
m_2=4/16
m_3= 6/16
m_4= 4/16
m_5= 1/16

M	m1	m2	m3	m4	m5	Total
P(z)	1/16	4/16	6/16	4/16	1/16	1

- DTE using Cumulative Distribution Function

According to the cumulative distribution function, we should get one for the total value of probability P(z) over all messages in seed space S to satisfy the properties of probability theory [10]. The following example describes that DTE using cumultative distribution function cannot satisfy the probability theory because the total values of probability P(z) of message in seed space S doesn't get one [15]. In this example we use five messages M= {m1,m2,m3,m4,m5} and calculate the probability P(z).

Number of message n= 5

For Distribution transforming encoder,

Seed space $S_m = 2^{n-1} = 2^{5-1} = 2^4 = 16$

By using cumultative distribution function

m1= 1/16 [(1,1)]

m2= 2/16 [(1,2),(2,1)]

m3= 2/16 [(1,3),(3,1)]

m4= 3/16 [(1,4),(4,1),(2,2)]

m5= 2/16 [(1,5),(5,1)]

M	m1	m2	m3	m4	m5	Total
P(z)	1/16	2/16	2/16	3/16	2/16	10/16

5. Results and Discussion

Many security areas such as web page and application using internet service used PBE for protection of their data from various attacks. However, PBE can't solve problem of the various security attacks especially brute force attack because most of the passwords are not much stronger. Our proposed honey encryption algorithm and honeywords generation methods can against brute force attack and many other attacks by deceiving the attacks. Moreover, our proposed method overcome the weakness of existing honey encryption and honeywords generations algorithm. The detail analysis and comparison results are shown in the following.

5.1 Implementation and Experimental Results

In this section we implement our proposed methodology in web application by developing a simulated messaging application and the language used is python programming language.

To evaluate the performance of our proposed honey encryption algorithm, we need to consider the false messages. The testing of the performance was conducted to distinguish between the correct key and incorrect keys. In this testing, we encode the message using the correct key. Then, we decrypt the message using the various incorrect keys. For all the testing, our proposed system generates the fake messages that are different from the original messages. The Figure 6 shows the original plaintext that was transmitted.

An attacker trying several keys is presented with the version of Figure 7 and Figure 8. There is no way the adversary can tell the original message from the fake message even if he acquires the original message.

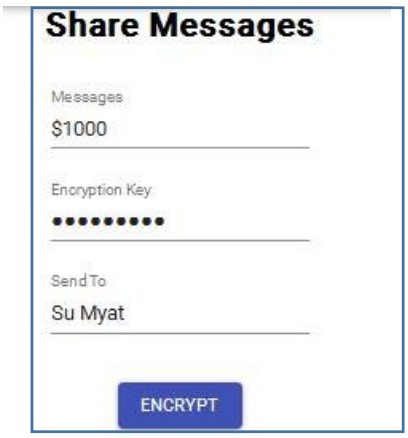

Figure 6: Original Plaintext Message

Figure 7: Intercepted Message acquired by Attacker using incorrect key

Figure 8: Intercepted Message acquired by Attacker using incorrect key

5.2 Comparative Analysis of Our Proposed System

Honey encryption (HE) algorithm is an interesting challenges algorithm and it plays an important role in various areas such as

business groups, banking system and computer users. Moreover, HE can protect from large number of attacks. Here it examines the comparative analysis of our proposed system and existing systems are described as follows.

1) Brute Force Attack

The web application is attacked by the attackers using the various attacks. Among them, the most dangerous attack for websites is brute force attack. The purpose of this attack is to steal the passwords and encryption key for obtaining the encrypted data. The attacker attacks the password file using the brute force attack and it finds the possible keys and password until the correct one is found. However, the attackers can't get the encryption key because of our proposed method. Although the password file in the database has been attacked by brute force attack, the attacker cannot classify which is real key or password if he gets password file. Moreover, attackers can take many times comparing into the existing security system because our system uses improved hashing and salting algorithm.

2) DoS Attack

The machines or network resources can be stopped and cann't be given any services to the users by the Denial of service (DoS) attack. Our proposed system is intended to reduce the DoS vulnerabilities. When a person tries to login into the system, the system automatically checks login of the user. If the attacker tries to login with his attacking password file, the system raises alarm for user and attack detected messages. And then, the system can block that particular attacker.

3) Dictionary Attack

Dictionary attack is one of the types of password guessing attacks and an attacker tries to login by cycling through combination of common words. Generally, this attack succeeds because most of the users choose weak or repeated passwords. However, our system can protect the passwords from the dictionary attack. When the attacker tries to login with randomly chooses an account with guessed password from password file, it can be two probabilities which are successful or fail. First is taking the password from the password file is correct and the next is getting the honeywords. If the attacker login with honeywords, the system raises an alarm for the user account and can detect this attack. Otherwise, if he enters with the real password easily, he can get the real message easily. But he cannot classify this message is real or fake messages.

4) DTE Process

The existing honey encryption algorithm and our proposed system denotes DTE as the main process. In this HE process, the message space that includes all plaintext messages is mapped into the seed space using DTE process. The existing DTE process uses cumulative distribution function for mapping the message into the seed space and it meets the message limitation problem. In our proposed system, DTE uses discrete distribution function and it can overcome the message limitation problem.

Table 3. Comparison Results of Existing Methods and Our Proposed Method

Techniques	Brute Force Attack	DoS Attack	Dictionary Attack	DTE Process
Password - based Encryption	Weak	Weak	Weak	-
Honey Encryption	Strong	Strong	Strong	Message Space Limitation
Our Proposed System	Strong	Strong	Strong	Overcome Message Space Limitation Problem

5.3 Comparison Results of Existing Honeywords Generation Methods and Our Proposed Honeywords Generation Method

In this section, we describe the comparison of the honeywords generation methods including our models in terms of flatness, typo safety and storage overhead.

1) Flatness

Our proposed system kept the list of s sweetwords that includes the real passwords and honeywords in the passwords files. There are two types of flatness such as perfectly-flat and approximately-flat for computing the security levels of honeywords generation methods. Depending on the types of flatness, we can decide which honeywords generation method is the better method for securing of our applications. Flatness computes that the attackers get how many times the real passwords. Let consider the attacker attacking the probability is w and this attacking probability compares the results of real password probability 1/s. When w>=1/s , the honeywords generation is approximately-flat and an adversary can guess the real password at random position. Otherwise, it said to be perfectly-flat if w<s [5].

Our proposed system is perfectly-flat and the attacker can't guess the real password position because we choose the honeyword from the other user's passwords. Our system has more advantages in security when the system includes more members.

2) Typo Safety

This is the important problem of existing honeywords generation algorithm. By creating the similar honeywords using honeywords generation algorithm, most of the users become typing mistake problem [5]. However, our proposed system can mostly reduce this typo safety problem comparing with the existing methods. Due to the using of other user's passwords as honeywords, the user can't miss with other user's passwords.

3) Storage Overhead

In the existing honeywords generation process, the algorithm generates at least twenty honeywords for deceiving the attackers. So, the attackers conflict and can't easily distinguish which one is real password or false password. But,

the system stores at least 21 passwords for one user in the database and the database becomes the storage overhead problem. In our proposed system, we denote s is sugarword that includes real password and honeywords. So, the honeywords become (s-1) because the user has one real password. Among the existing system, paired distance protocol (PDP) method is the best reducing storage overhead problem and it stores only random string of honeywords without storing the whole honeywords [7]. However, our proposed method can mostly reduce the storage overhead than the PDP algorithm because our system uses the indexes of member's password as honeyindexes.

Table 4. Comparison Results of Existing Honeywords Generation Methods and Our Proposed Method

No	Methods	Flatness	Typo Safety	Storage Overhead
1	CTD	1/k	Low	(s-1)*n
2	Password Model	1/k	High	(s-1)*n
3	Take a Tail	1/k	High	(s-1)*n
4	PDP	1/k	High	(1+RS)*n
5	Our Method	1/k	High	(1*n)

CTD- Chaffing by Tweaking

PDP- Paired Distance Protocol

s- Number of sweetwords in password file

n- Number of users

RS- Random String

5.4 Time Comparison Results of MD5 and Our Proposed Hashing Algorithm

Our proposed honey encryption process are designed by python programming language. This python programming language is flexible for all real world system and easy for the programmers to study. In this paper, we proposed three main contributions: improved DTE process using discrete distribution function, new honeywords generation method and the last is improved hashing algorithm for providing the better security of our proposed system with less time complexity comparing to the existing MD5 hashing algorithm. From the experimental results, the MD5 hashing algorithm takes more time than our proposed algorithm for converting the passwords into the hash code because our proposed system use simple binary operation.

Table5. Time Comparison Results of Existing MD5 Hashing Algorithm and Our Proposed Algorithm

		Time Complexity (Milliseconds)			
Length of Passwords (in character)		6	8	10	12
Methods	MD5	1740	2854	3260	3681
	Our Proposed Algorithm	31	46	62	78

In the following table, we uses the user's passwords such as "123456, hello123, student321, December3040" for the comparing the time results of other proposed hashing algorithm and existing MD5 hashing algorithm.

6. Conclusions

Most of the organizations try to against the brute force attack for protecting their private data. Honeywords generation algorithm can make the confusion for the attackers by storing the real password with the false passwords or honeywords in the password files. However, honeywords generation algorithm can meet the storage overhead problem if many of the users use this application. Honey encryption algorithm can protect the brute force attack by producing honey messages if the attackers use the honeywords. The existing honey encryption algorithm has the message space limitation problem and it can't send more than four messages by using cumulative distribution function. Our proposed honeywords generation algorithm and honey encryption algorithm can solve the storage overhead problem and message space limitation problem. Moreover, we proposed new hashing algorithm for securing our system and for getting the faster processing time compared to the existing MD5 hashing system.

References

[1] J.H. Saltzer, D. P. Reed and D.D. Clark, "Review for Paper: End to End Arguments in System Design" Polytechnic University, USA, 2006.

[2] K. S. M. Moe, T. Win, "Enhanced Honey Encryption Algorithm for Increasing Message Space against Brute Force Attack" in 15th Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI) Conference, Thailand, Chaing Rai, 2018.

[3] A. Juels and T. Ristenpart, "Honey encryption: security beyond the brute-force bound," in Advances in Cryptology EUROCRYPT 2014, Springer, Berlin, Germany, 2014, vol. 8441 of Lecture Notes in Computer Science, pp. 293–310.

[4] N.Tyagi, J. Wang, K. Wen and D. Zuo, Honey Encryption Application, Computer and network Security, Springer, 2015.

[5] D. Gross, "50 million compromised in Evernote hack" CNN, 4 March 2013.

[6] M. Weir, S. Aggarwal, B. de Medeiros, and B. Glodek, "Password cracking using probabilistic context-free grammars," In IEEE Symposium on Security and Privacy (SP), pp-162-175, 2009.

[7] P.G. Kelley, S. Komanduri, M.L. Mazurek, R. Shay, T. Vidas, L. Bauer, N. Christin, L.F. Cranor, and J. Lopez, "Guess again (and again and again): Measuring password strength by simulating password- cracking algorithms," In IEEE Symposium on Security and Privacy (SP), pp- 523-537, 2012.

[8] M. Bakker and R. V.D .Jagt, "GPU-based password cracking. Technical report," Univ. of Amsterdam, 2010.

[9] S. Kharod, N. Sharma and A. Sharma, "An Improved Hashing Based Password Security Scheme Using Salting and Differential Masking, in 2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Direction)," India, 2018.

[10] A. Juels and R. L. Rivest, "Honeywords Making Password Cracking Detectable" in ACM SIGSAC Conference on Computer & Communications Security, ser. CCS '13. New York, NY, USA: ACM, 2013, pp. 145–160.

[11] Z. A. Genc, S.Kardas and M.S. Kiraz, "Examination of a New Defense Mechanism: Honeywords," in 11th WISTP International Conference on Information Security Theory and Practice, Heraklion, Crete, Greece, 2017.

[12] N. Chakraborty and S. Mondal, "A New Optimized Honeyword Generation Approach for Enhancing Security and Usability," arXiv preprint arXiv: 1509.06094v1, 2015.

[13] S. R. Shinge and R. Patil, "An Encryption Algorithm based on ASCII Value of Data" International Journal of Computer Science and Information Technologies, vol. 5(6), 2014, pp-7232-7234.

[14] A. Juels and R. L. Revist, "Honeywords: Making Password Cracking Detectable" In MIT CSAIL, 2013.

[15] B. Addanki, Computer Physicist, "What is the difference between a probability density function and cumulative distribution function," 2015. http:// www.quora.com.