Scientific Research Publishing

Journal of Computer and Communications, 2022, 10, 22-36
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

A Novel Hybrid Encryption Method Based on Honey Encryption and Advanced DNA Encoding Scheme in Key Generation

Nwe Ni Khin, Thanda Win

Department of Computer Engineering and Information Technology, Yangon Technological University, Yangon, Myanmar
Email: nnkhin13itphd@ytu.edu.mm, dr.thandawin@ytu.edu.mm

How to cite this paper: Khin, N.N. and Win, T. (2022) A Novel Hybrid Encryption Method Based on Honey Encryption and Advanced DNA Encoding Scheme in Key Generation. *Journal of Computer and Communications*, **10**, 22-36.
https://doi.org/10.4236/jcc.2022.109002

Received: July 25, 2022
Accepted: September 6, 2022
Published: September 9, 2022

Abstract

Nowadays, increased information capacity and transmission processes make information security a difficult problem. As a result, most researchers employ encryption and decryption algorithms to enhance information security domains. As it progresses, new encryption methods are being used for information security. In this paper, a hybrid encryption algorithm that combines the honey encryption algorithm and an advanced DNA encoding scheme in key generation is presented. Deoxyribonucleic Acid (DNA) achieves maximal protection and powerful security with high capacity and low modification rate, it is currently being investigated as a potential carrier for information security. Honey Encryption (HE) is an important encryption method for security systems and can strongly prevent brute force attacks. However, the traditional honeyword encryption has a message space limitation problem in the message distribution process. Therefore, we use an improved honey encryption algorithm in our proposed system. By combining the benefits of the DNA-based encoding algorithm with the improved Honey encryption algorithm, a new hybrid method is created in the proposed system. In this paper, five different lookup tables are created in the DNA encoding scheme in key generation. The improved Honey encryption algorithm based on the DNA encoding scheme in key generation is discussed in detail. The passwords are generated as the keys by using the DNA methods based on five different lookup tables, and the disease names are the input messages that are encoded by using the honey encryption process. This hybrid method can reduce the storage overhead problem in the DNA method by applying the five different lookup tables and can reduce time complexity in the existing honey encryption process.

Keywords

Honey Encryption, DNA Encoding, Hybrid Method, Data Lookup Tables,

Time Complexity, Storage Overhead

1. Introduction

Using cryptographic methods, valuable information is kept secure. By using such methods, the sender can securely transmit and store sensitive information over the internet. When information is encrypted using a cryptographic method, the resultant output may be unintelligible to an outsider without access to the key. Knowing the key is a crucial component of the encryption and decryption processes [1]. Nowadays, most web applications generate the key from the user's entered password because it can be easily remembered by the users. Therefore, it can become the main weakness of the message transmission process because security flaws demonstrated that many people all across the world utilize quick-to-remember passwords. The majority of people just choose one simple password that they use for all websites, writing it down for future use, making it more vulnerable to attacks.

The honey encryption is the data protection algorithm for protecting the brute force attack and it can deceive the attackers. If an attacker attempts to decrypt plaintext with the incorrect key or honeywords, a user data protection mechanism known as honey encryption (HE) can produce valid-looking plaintext and can effectively fool unauthorized users. If an attacker attempts to decrypt with any of a large number of invalid passwords, the honey encryption process generates a honey message. If not, the HE procedure generates the right ciphertext. Every incorrect password guess made by a hacker is made into a perplexing dead end by HE [2].

The Distribution Transforming Encoder is a cryptographic primitive used in the creation of the improved HE processes. The DTE is a collection of encoding and decoding operations, each of which produces a value in the seed space S of n-bit strings as an output and accepts a space of plaintext messages as input (M). Values in the seed space S of n-bit strings are converted into plaintext during the decoding phase. In order to determine the appropriate message ratio, the DTE method in the improved HE algorithm uses the probability distribution function [3].

The example of an improved enhanced honey encryption process is shown in **Figure 1**. The improved honey encryption process mainly consisted of two main parts: the message distribution process and the passwords or key distribution process. The key distribution process used the simple hashing and salting algorithm for encoding the passwords [3]. In our proposed system, we use DNA encoding in the password encoding process to reduce the time complexity problem and improve the security level of the key process. However, the existing DNA encryption has complex steps and memory overhead problems. Therefore, we proposed an advanced DNA encoding scheme that uses five different lookup

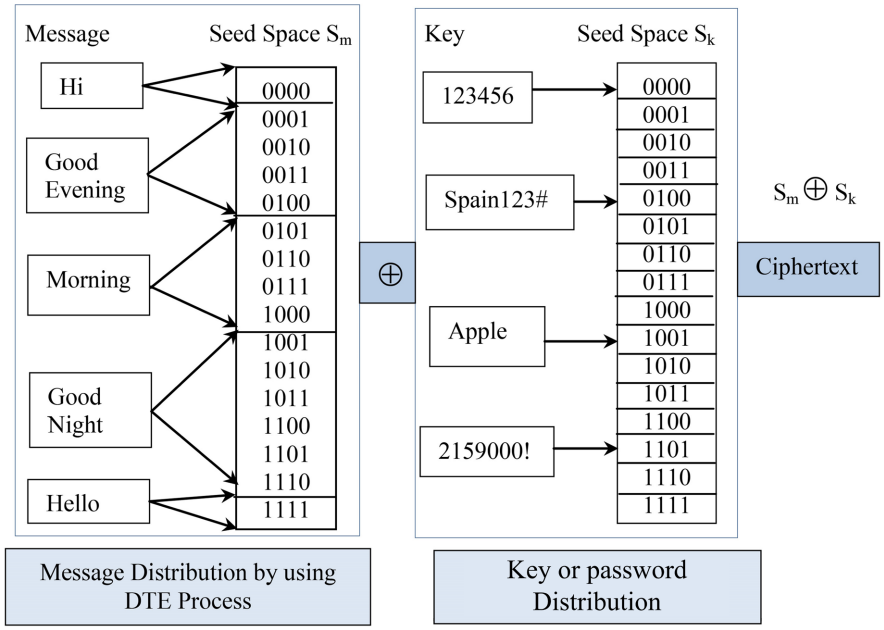

Figure 1. Honey encryption process using hashing algorithm [3].

tables for the key generation process in an improved honey encryption algorithm.

2. Related Works

Ari, J. and Thomas, R. created the honey encryption (HE) algorithm, a new encryption technique to guard against brute force attacks. Because HE encrypted private messages using the user's credentials, it was comparable to the password-based encryption algorithm (PBE). In order for HE to effectively defended against a brute force attack. The distribution transforming encoder (DTE) was the primary component of HE, and HE can be applied to a variety of diverse applications, including the storage of genetic data and credit card numbers [4] [5]. HE had limitations in the DTE process for inserting the messages into the seed area, although it protected against brute force attacks.

The improved hashing and honey-based stronger password prevention against brute force attacks proposed by Moe, K. S. M., and Win, T. [6] solved the typo error by using the passwords of other users as honeywords in the database instead of creating honeywords, reduced storage space, and used unique hashing algorithm that saved a significant amount of time. The method took less time to execute than the MD5 algorithm.

In 2019 [7], Bhavani, Y., Puppala, S., Krishna, B., J., and Madarapu, S. showed evidence that DNA was altered using AES. The system had gone through multiple steps, first converting the user's inputted data (plaintext) to Hexadecimal format. From there, binary data was converted to DNA code. Although brute force assaults were not protected, this system took a number of procedures to thwart differential and linear cryptanalysis attempts.

Pushpa, B. R. [8] proposed a new technique for data encryption using the DNA sequence system. The defense was more effective against some assaults. However, ASCII's greater key length led to memory overload.

Therefore, we use an improved honey encryption algorithm to overcome the message limitation in our proposed system. This honeywords generation system can reduce the drawbacks of the existing honeywords generation algorithms, such as reducing storage costs. Moreover, it can easily overcome the typo safety problem. For securing password files and reducing computation time, we use enhanced DNA encoding in the key distribution process. The enhanced DNA encoding is faster than the existing hashing and salting algorithms in the key generation process. Finally, our proposed system can overcome message limitations and reduce computation time.

3. Background Theory

Information security plays a very important role in every aspect of life. It provides more security for information that cannot be hacked by anyone but the sender and receiver. The proposed approach will provide more security to the data while transmitting the messages by using improved honey encryption with an advanced DNA encoding scheme in the key generation process.

3.1. Improved Honey Encryption Algorithm

The honey encryption process includes honeywords generation process for key or passwords distribution and distribution transforming encoding (DTE) process for message distribution.

In **Figure 2**, the message space M consists of five messages (m), such as "Apple, Orange, Strawberry, Chocolate, and Mango". This message space is mapped to the seed space, S_m using the DTE process. The DTE process uses a discrete distribution function for mapping the message space to the seed space S_m overcoming the message space limitation. In the key or password distribution module, keys or passwords are mapped randomly into seed space S_k. Before mapping the key, passwords are converted into those keys using advanced DNA encoding. To generate the ciphertext, the resulting seed space S_k is XORed with seed space of message S_m.

3.2. Distribution Transforming Encoder

Distribution transforming encoder (DTE), the primary component of improved HE, is capable of both encoding and decoding. The message space, including all plaintext messages, is what the DTE encode accepts as input m€ M, utilizing the discrete distribution function for mapping the seed space S_m [3] in the message distribution process. The DTE process uses the discrete distribution function for mapping the messages in the seed space. In **Figure 3**, we map the five messages by calculating the following step-by-step process, and we denote n as the total number of messages.

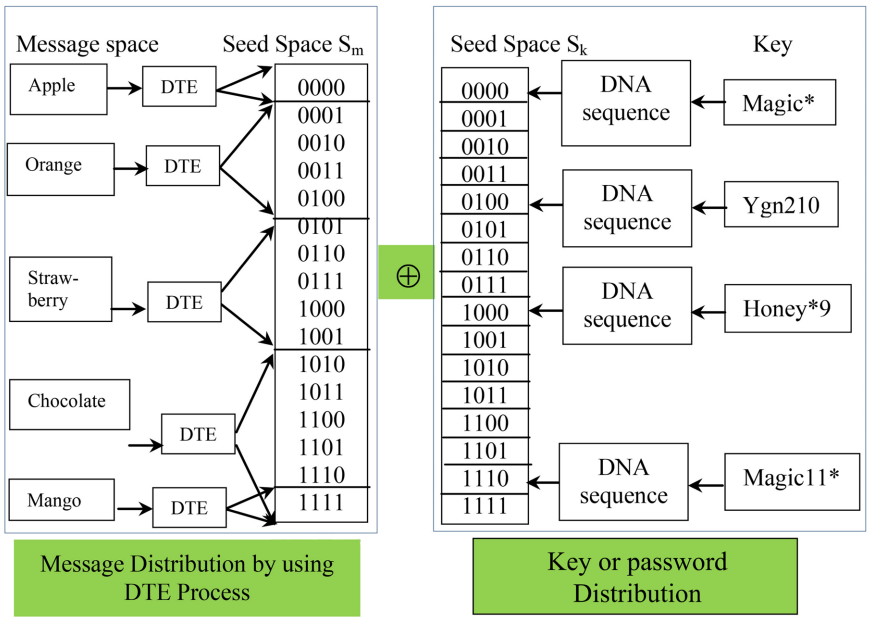

Figure 2. Process of improved honey encryption algorithm using DNA encoding.

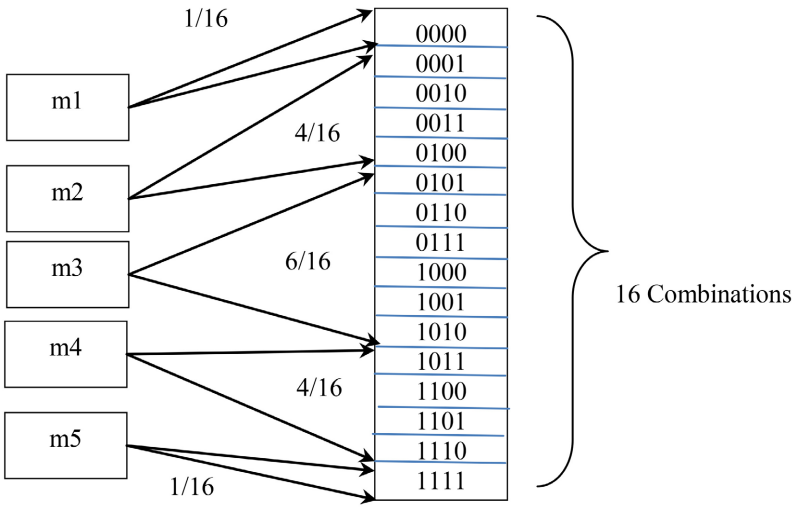

Figure 3. Range of seed space S_m (m1 = 1/6; m2 = 4/16; m3 = 6/16; m4 = 4/16; m5 = 1/16).

Step 1: Number of input messages: 5.

Step 2: Using 2^{n-1} formula, calculate the seed space S_m range.

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

Step 3: Accordingly, to the formula, we take 4 bits binary numbers for placing the seed space.

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

Step 4: By using the Discrete Distribution Function, map the messages to the following range of seed space, S_m.

From the calculation, we can achieve a better result in the HE algorithm if we use the DTE process using a discrete distribution function.

3.3. Advanced DNA Encoding Scheme

Deoxyribonucleic Acid (DNA) is made up of four basic nucleic acids: adenine (A), cytosine (C), guanine (G), and thymine (T). It transfers the characters to the DNA-based entities and converts the user's password into the DNA code.

This DNA code is used as the input in the password distribution process. The second approach generates the DNA code that can be used as the key to the HE processes. Therefore, encryption and decryption can be done using five data lookup tables. Figure 4 is the block diagram of the DNA encoding process.

In our proposed system, we use proposed DNA encryption process in the key generation process from user passwords. In the following Figure 5, the user's chosen password is "Crypt@4U". In the DNA encryption process, user-selected passwords are converted into the DNA sequence using the randomly chosen five data lookup tables.

3.3.1. Creating Five Different Data Lookup Tables

We use 64 characters in creating five different data lookup tables. We don't need to consider case-sensitive in our creation of data lookup tables. Therefore, we can reduce the memory overload problem compared to the conventional DNA encryption algorithm because the conventional DNA encryption algorithm includes the complex steps. We arrange the following five different data lookup

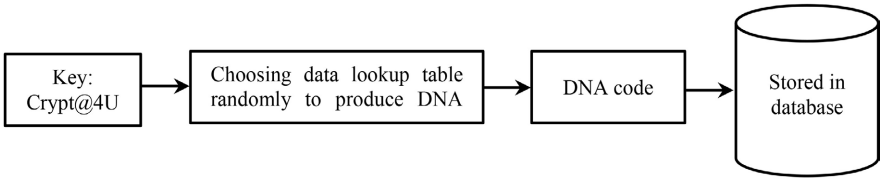

Figure 4. DNA encoding process.

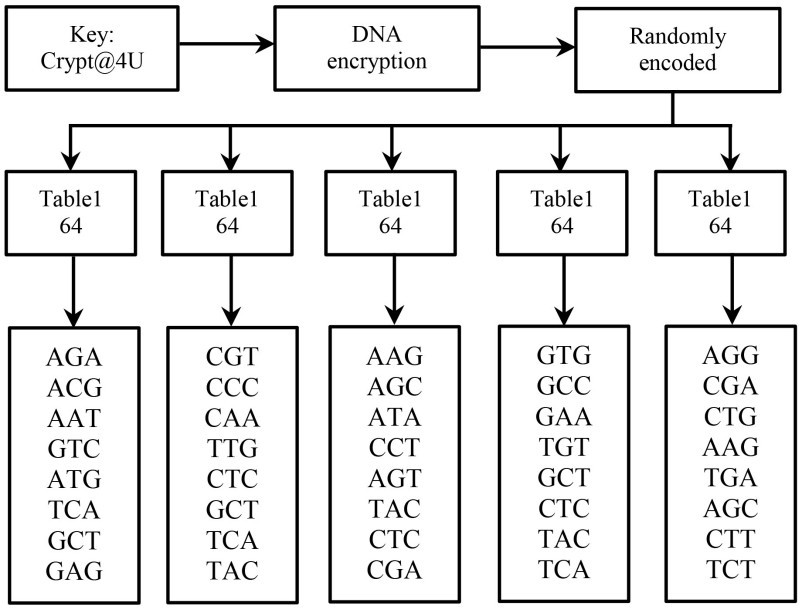

Figure 5. Enhanced DNA encryption.

tables: The data lookup table-1 is arranged row-by row from top to bottom; the data lookup table-2 is arranged row-by-row from bottom to top; the data lookup table-3 is arranged by column to column from left to right; the data lookup table-4 is arranged by column to column from right to left; and finally, diagonal arrangement settings. The five different data lookup tables are described as Tables 1-5.

Table 1. Data lookup table 1.

A = AAA	B = ACA	C = AGA	D = ATA	E = GAA	F = GCA	G = GGA	H = GTA
I = AAC	J = ACC	K = AGC	L = ATC	M = GAC	N = GCC	O = GGC	P = GTC
Q = AAG	R = ACG	S = AGG	T = ATG	U = GAG	V = GCG	W = GGG	X = GTG
Y = AAT	Z = ACT	1 = AGT	2 = ATT	3 = GAT	4 = GCT	5 = GGT	6 = GTT
7 = CAA	8 = CCA	9 = CGA	0 = CTA	! = TAA	@ = TCA	# = TGA	$ = TTA
* = CAC	? = CCC	/ = CGC	> = CTC	< = TAC	~ = TCC	Space = TGC	| = TTC
\\ = CAG	_ = CCG	= = CGG	+ = CTG	- = TAG	, = TCG	. = TGG	: = TTG
; = CAT	% = CCT	& = CGT	^ = CTT	(= TAT	) = TCT	[= TGT	] = TTT

Table 2. Data lookup table 2.

; = AAA	% = ACA	& = AGA	^ = ATA	(= GAA	) = GCA	[= GGA	] = GTA
\\ = AAC	_ = ACC	= = AGC	+ = ATC	- = GAC	, = GCC	. = GGC	: = GTC
* = AAG	? = ACG	/ = AGG	> = ATG	< = GAG	~ = GCG	Space = GGG	| = GTG
7 = AAT	8 = ACT	9 = AGT	0 = ATT	! = GAT	@ = GCT	# = GGT	$ = GTT
Y = CAA	Z = CCA	1 = CGA	2 = CTA	3 = TAA	4 = TCA	5 = TGA	6 = TTA
Q = CAC	R = CCC	S = CGC	T = CTC	U = TAC	V = TCC	W = TGC	X = TTC
I = CAG	J = CCG	K = CGG	L = CTG	M = TAG	N = TCG	O = TGG	P = TTG
A = CAT	B = CCT	C = CGT	D = CTT	E = TAT	F = TCT	G = TGT	H = TTT

Table 3. Data lookup table 3.

A = AAA	I = ACA	Q = AGA	Y = ATA	7 = GAA	* = GCA	\\ = GGA	; = GTA
B = AAC	J = ACC	R = AGC	Z = ATC	8 = GAC	? = GCC	_ = GGC	% = GTC
C = AAG	K = ACG	S = AGG	1 = ATG	9 = GAG	/ = GCG	= = GGG	& = GTG
D = AAT	L = ACT	T = AGT	2 = ATT	0 = GAT	> = GCT	+ = GGT	^ = GTT
E = CAA	M = CCA	U = CGA	3 = CTA	! = TAA	< = TCA	- = TGA	(= TTA
F = CAC	N = CCC	V = CGC	4 = CTC	@ = TAC	~ = TCC	, = TGC	) = TTC
G = CAG	O = CCG	W = CGG	5 = CTG	# = TAG	Space = TCG	. = TGG	[= TTG
H = CAT	P = CCT	X = CGT	6 = CTT	$ = TAT	| = TCT	: = TGT	] = TTT

Table 4. Data lookup table 4.

; = AAA	\\ = ACA	* = AGA	7 = ATA	Y = GAA	Q = GCA	I = GGA	A = GTA
% = AAC	_ = ACC	? = AGC	8 = ATC	Z = GAC	R = GCC	J = GGC	B = GTC

Continued

& = AAG	= = ACG	/ = AGG	9 = ATG	1 = GAG	S = GCG	K = GGG	C = GTG
^ = AAT	+ = ACT	> = AGT	0 = ATT	2 = GAT	T = GCT	L = GGT	D = GTT
(= CAA	- = CCA	< = CGA	! = CTA	3 = TAA	U = TCA	M = TGA	E = TTA
) = CAC	, = CCC	~ = CGC	@ = CTC	4 = TAC	V = TCC	N = TGC	F = TTC
[= CAG	. = CCG	Space = CGG	# = CTG	5 = TAG	W = TCG	O = TGG	G = TTG
] = CAT	: = CCT	| = CGT	$ = CTT	6 = TAT	X = TCT	P = TGT	H = TTT

Table 5. Data lookup table 5.

A = AAA	! = ACA	> = AGA	_ = ATA	. = GAA	& = GCA	) = GGA	] = GTA
I = AAC	B = ACC	@ = AGC	< = ATC	= = GAC	: = GCC	^ = GGC	[= GTC
P = AAG	J = ACG	C = AGG	# = ATG	~ = GAG	+ = GCG	; = GGG	(= GTG
V = AAT	Q = ACT	K = AGT	D = ATT	$ = GAT	Space = GCT	- = GGT	% = GTT
1 = CAA	W = CCA	R = CGA	L = CTA	E = TAA	* = TCA	! = TGA	, = TTA
5 = CAC	2 = CCC	X = CGC	S = CTC	M = TAC	F = TCC	? = TGC	\\ = TTC
8 = CAG	6 = CCG	3 = CGG	Y = CTG	T = TAG	N = TCG	G = TGG	/ = TTG
0 = CAT	9 = CCT	7 = CGT	4 = CTT	Z = TAT	U = TCT	O = TGT	H = TTT

3.3.2. Case Study

The following example shows the detailed case study process of the advanced DNA encryption algorithm using five different data lookup tables. In this case study, "Crypt@4U" is used as an example password for converting the DNA code using the advanced DNA encryption algorithm.

Password chosen by the user: Crypt@4U

Convert the user's password into DNA code

Using data lookup table 1

Crypt@4U-AGAACGAATGTCATGTCAGCTGAG

Using data lookup table 2

Crypt@4U-CGTCCCCAATTGCTCGCTTCATAC

Using data lookup table 3

Crypt@4U-AAGAGCATACCTAGTTACCTCCGA

Using data lookup table 4

Crypt@4U-GTGGCCGAATGTGCTCTCTACTCA

Using data lookup table 5

Crypt@4U-AGGCGACTGAAGTGAAGCCTTTCT

Among the five different data lookup tables, the algorithm randomly chooses one data lookup table. Therefore, the resulting DNA codes are different, although the passwords are the same. So, our advanced DNA encryption algorithm can protect against brute force attacks.

4. Flowchart of Our Proposed System

The flowchart of our proposed system is shown in Figure 6. This flowchart illustrates the system's step-by-step processes.

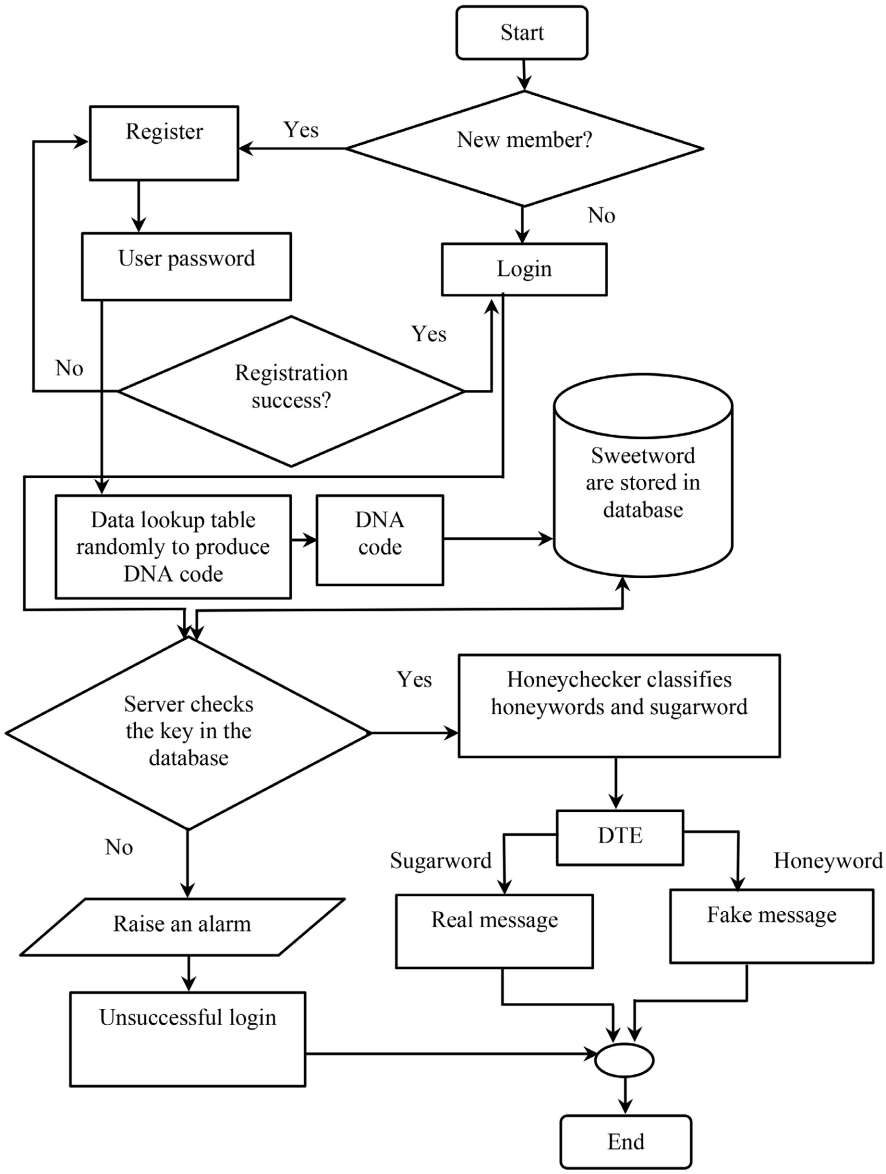

Figure 6. Flowchart of our proposed system.

The server determines during this procedure whether the password or username disappears from the database if either of the users goes through the login process. In our proposed system, there are two portions in the login process: new members and existing members. Firstly, if the login process fails, the user needs to go through the registration process when the user is not a member of the system. When the user is an existing member, the user needs to fill in the username and password in the login process after the registration process. The system converts the passwords into the DNA code using an advanced DNA encryption algorithm for storing them in the database as the code. Otherwise, the user can enter the system using their username and password. The system checks whether the username and password exist in the database or not. If the username and password do not exist in the database, the system sends an alarm message "key

was wrong" and the user receives an "unsuccessful login". If the username and password exist in the database, the system sends password to the honeychecker for determining of honeywords and real password (sugarword). Finally, the user performs the DTE process if the password is either honeywords or a real password. If the password is a real one, the user gets the real message. Otherwise, the user can get the fake message if the password is honeyword.

5. Results and Discussions

In this section, we discussed two main parts: the implementation process and experimental results.

5.1. Implementation Process

The implementation of the proposed system includes a registration page, login page, home page, inbox page, and message encryption and decryption process. If the user is a new user, the user must fill out the user registration form and complete the registration process. Otherwise, users can log in to the system using the login button. After successful login, the user can see the home page of their accounts. On their home page, users can initiate two processes, such as message encryption and decryption. In the encryption process, users type the message and receiver name and then click the encryption button. The following Figure 7 and Figure 8 describe the receiver decrypting the message using the correct key and getting the correct message. Otherwise, he decrypts the wrong key and gets the wrong message.

5.2. Experimental Results

The proposed systems are implemented in a web application by developing a simulated messaging application and the language used is the Python programming language. In this testing, we calculated the execution time for the encryption

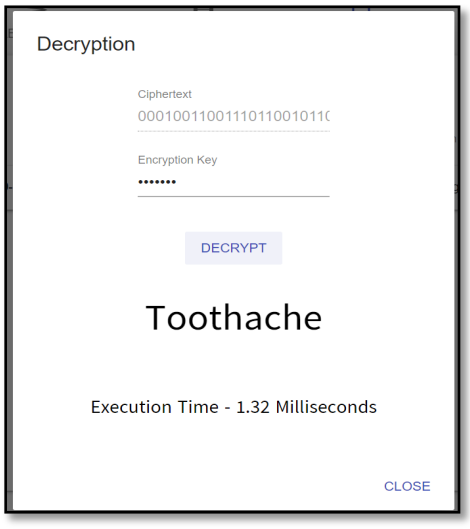

Figure 7. Decryption testing with correct key.

Figure 8. Decryption testing with wrong key or honeyword.

of the same message (Typhoid) with the same password (smile@9) using different data lookup tables. The execution times are varied depending on the data lookup table distribution and results are as shown in **Table 6**. The data distributed column to column from left to right in the data lookup table-3 is the most time-consuming, and the data distributed row by row from top to bottom in the data lookup table-1 is the least time-consuming according to the testing-1.

The following **Table 7** describes the execution times for the different messages "Omicron, Coronavirus, Monkeypox, and InfluenzaA (H1N1)" encrypted with the same password "blockC!" using the different data lookup tables. Because the "Influenza" message was used repeatedly in previous testing [9], the execution time for the 16 character message "InfluenzaA (H1N1)" encryption is 1.54 ms, which is less than the execution time for the "7, 9, and 11" character messages encryption. So, our proposed system can reduce the time when the same messages are transferred repeatedly.

The next testing 3 is to encrypt the different messages with different passwords using the random different data lookup tables. **Table 8** shows the execution times for different messages of the same length "Lassa fever, Ebola virus, and YellowFever" encrypted with 7 passwords, 10 passwords, and 15 passwords, respectively. The execution time is 1.26 ms, 1.28 ms, and 1.38 ms, respectively. In the testing 4, the 18 characters message "Lassa fever-Guinea" encryption with 15 passwords has an execution time of 1.37 ms as shown in **Table 9**. So, the password length is the same, the execution time is also nearly the same according to the testing, 3 and 4.

In the final testing, the different password lengths (8, 10 and 12) were encrypted compared with the other generations of honeywords using a hashing algorithm and our proposed models using a DNA encoding scheme. It can be seen that the proposed method, which encrypts the messages that are the most commonly used passwords by users, the time complexity of the proposed method is faster than the existing method [3], as shown in **Table 10**.

Table 6. Same messages encryption with same passwords using five data lookup tables.

Testing1	Same Passwords	Lookup Table	Execution Time (ms)
Typhoid	smile@9	1	1.70
Typhoid	smile@9	2	1.78
Typhoid	smile@9	3	1.88
Typhoid	smile@9	4	1.81
Typhoid	smile@9	5	1.85

Table 7. Different messages encryption with same passwords using five data lookup tables.

Testing2	Same Passwords	Lookup Table	Execution Time (ms)
Omicron	block!C	1	1.82
Coronavirus	block!C	2	1.75
Monkeypox	block!C	3	1.56
InfluenzaA(H1N1)	block!C	4	1.54

Table 8. Different messages (11 characters) encryption with Different passwords using random different data lookup tables.

Testing3	Different Passwords	Execution Time (ms)
Lassa Fever	Biebar*	1.26
Ebola Virus	David(123)	1.28
Yellow Fever	Aniston558/cele	1.38

Table 9. Different messages (18 characters) encryption with Different passwords using random different data lookup tables.

Testing4	Different Passwords (15 characters)	Execution Time (ms)
Lassa fever-Guinea	taylorswift*cel	1.37

Table 10. Time comparison results between existing algorithm and our proposed system.

Passwords Length (Characters)	Time Complexity (ms)		
	8	10	12
Hashing and Salting Algorithm	15.6	15.6	15.6
Our Proposed System	1.78	1.83	1.97

According to Table 10, we make the testing process of the length of the passwords longer by using the hashing and salting algorithm and our proposed system. From the results of the existing and our proposed systems, it is clear that our proposed system takes less time for encryption and decryption processes compared to the existing system. Therefore, the processing time of our proposed system is faster than the existing hashing and salting algorithm because the ex-

isting system has complex steps that include left shift, right shift, and binary conversion. However, our system uses five simple data lookup tables.

5.3. Comparison Results of Mathematical Model between Traditional Honey Encryption Algorithm and Improved Honey Encryption Algorithm

The main creation of the honey encryption algorithm is the discrete transforming encoder (DTE). The purpose of DTE is to place the message M into the seed space S_m for distribution. The DTE uses the probability function, and it needs to satisfy the property of probability theory. The properties of probability are that the range of values exists between 0 and 1, and the sum of total values needs to get the value of 1 [10]. The improved honey encryption algorithm uses the discrete distribution function of the probability function to overcome the message space limitation problem. The traditional honeyword encryption algorithm uses the cumulative distribution function, and it has the message space limitation problem because it can't satisfy the property of probability theory. The following examples show that the improved honey encryption algorithm satisfies the properties of probability theory and can overcome the message limitation problems. In the example, we define the seed space message as S_m, and the seed space value range P(s) is between 0 and 1, and M represents the message.

From Table 11, the traditional honey encryption algorithm using the cumulative distribution function can't satisfy the probability theory. Therefore, the traditional method faces the message space limitation problem when the number of messages is greater than five. However, the improved honey encryption algorithm can solve the message space limitation problem because it can satisfy the property of probability theory [3]. Therefore, we use an improved honey encryption algorithm in our proposed system.

5.4. Limitations

Our proposed system is tested with the disease names as the input messages and it can use up to 30 characters for the message space, but the system is limited to a maximum of 255 characters. The testing results are listed by disease name, and every test is limited to test by disease name because proposed system is based on disease name. Since most users only use up to 30 characters, system tested up to 30 passwords in our proposed system. A limitation of the study in this system is that users can securely transfer personal information to each other over the web via a single Wi-Fi network.

Table 11. Testing results between improved honey encryption algorithm and traditional honey encryption algorithm.

Message (M)	The range of seed space value P(s)					
	m1	m2	m3	m4	m5	Total
Traditional honey encryption algorithm	1/16	2/16	2/16	3/16	2/16	10/16
Improved honey encryption algorithm	1/16	4/16	6/16	4/16	1/16	1

6. Conclusion

The traditional honey encryption algorithm is the strong encryption algorithm for protecting against brute force attacks and facing the message space limitation problem. The improved honey encryption algorithm can solve the message space limitation problem. So, we use a hybrid method of the improved honey encryption and the enhanced DNA encoding scheme in our proposed system. Therefore, our proposed system can protect against brute force attacks and can solve the message space limitation problem. Moreover, we use the enhanced DNA encoding scheme in the key generation process of the improved honey encryption algorithm. Therefore, our proposed system can reduce the processing time more than the traditional honey encryption algorithm. And then, if we apply the advanced DNA encoding scheme in the honeyword generation process, our proposed system can get better security. In the paper, we describe the details and case study of an advanced DNA encoding scheme and mathematical model in order to prove that the DTE process can overcome the message space limitation problem. In the future, we will apply our proposed improved honey encryption algorithm that uses an advanced DNA encoding scheme in the key generation process in large organizations, such as the area of medical fields for transferring medical records.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Hassan, A.M., Meshrif, A., Osama, R.S. and Khalid, A. (2019) Design and Analysis of DNA Encryption and Decryption Technique based on Asymmetric Cryptography System. *International Journal of Advanced Computer Science and Applications*, **10**. http://b7a1f096cdaace21c6a3680c62af103c475c.pdf

[2] Nirvan, T., Jessica, W., Kevin, W. and Daniel, Z. (2015) Honey Encryption Application. https://courses.csail.mit.edu/6.857/2016/files/tyagi-wang-wen-zuo.pdf

[3] Khin, S.M.M. and Thanda, W. (2018) Enhanced Honey Encryption Algorithm for Increasing Messages Space against Brute Force Attack. 15*th International Conference on Electrical Engineering/Electronics*, *Computer*, *Telecommunications and Information Technology*, 86-89.

[4] Ari, J. and Thomas, R. (2014) Honey Encryption: Security beyond the Brute-Force Bound. In: Nguyen, P.Q. and Oswald, E., Eds., *Advances in Cryptology—EUROCRYPT* 2014, *Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, 293-310. https://doi.org/10.1007/978-3-642-55220-5_17

[5] Zhicong, H., Erman, A., Jacques, F., Jean-Pierre, H. and Ari, J. (2015) Genoguard: Protecting Genomic Data against Brute-Force Attacks. 2015 *IEEE Symposium on Security and Privacy*, San Jose, 17-21 May 2015, 447-462.

[6] Khin, S.S.M. and Thanda, W. (2017) Improved Hashing and Honey-Based Stronger Password Prevention against Brute Force Attack. 2017 *International Symposium on Electronics and Smart Devices*, Yogyakarta, 17-19 October 2017, 1-5.

[7] Bhavani, Y., Puppala, S., Krishna, B.J. and Madarapu, S. (2019) Modified AES Using Dynamic S-Box and DNA Cryptography. 2019 *International Conference on I-SMAC* (*IoT in Social*, *Media*, *Analytics and Cloud*), Palladam, 12-14 December 2019, 164-168. https://doi.org/10.1109/I-SMAC47947.2019.9032516

[8] Pushpa, B.R. (2017) A New Technique for Data Encryption using DNA Sequence, 2017 *International Conference on Intelligent Computing and Control*, Coimbatore, 23-24 June 2017, 1-4. https://doi.org/10.1109/I2C2.2017.8321834

[9] Nwe, N.K. and Thanda, W. (2022) Enhanced DNA Encoding Scheme in Honey Encryption. *International Journal of Computer* (*IJC*), **42**, 91-104.

[10] Addanki, B. (2015) What Is the Difference between a Probability Density Function and Cumulative Distribution Function. http://www.quora.com