Indonesian Journal of Electrical Engineering and Computer Science
Vol. 13, No. 2, February 2019, pp. 649~656
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v13.i2.pp649-656

A comprehensive review of honey encryption scheme

Abiodun Esther Omolara[1], Aman Jantan[2], Oludare Isaac Abiodun[3]
[1,2]School of Computer Sciences, Universiti Sains Malaysia, Penang, Malaysia
[3]Department of Computer Science, Bingham University, Karu, Nigeria

Article Info

Article history:

Received Jul 12, 2018
Revised Nov 27, 2018
Accepted Dec 27, 2018

Keywords:

Brute-force
Decoys
Distribution transforming encoder (DTE)
Honey encryption
Password-based encryption

ABSTRACT

We present a comprehensive survey of the Honey Encryption (HE) scheme. Honey Encryption is an encryption scheme that provides resilience against brute-force attack by serving up plausible-looking but fake plaintext for every invalid key used by an intruder to decrypt a message. Our goal is to furnish researchers with the framework of the scheme not just for implementation purpose but to identify the gaps in the scheme and answer the open questions that remain unanswered by the small set of research carried out since its inception. We identified two major open areas which are the difficulty of creating semantically and contextually plausible-looking and convincing decoy message that is good enough to fool the attacker into believing he has the original message. Secondly, typo problem; where a fake plaintext appears valid to a legitimate user when he mistakenly enters a wrong key. Our findings consolidate the need for further research as state-of-the-art research fails to produce convincing decoys that are good enough to keep the attacker from acquiring the message.

Corresponding Author:

Aman Jantan,
School of Computer Sciences,
Universiti Sains Malaysia, Penang.
Email: aman@usm.my, styleest2011@gmail.com

1. INTRODUCTION

The craft of deception is indispensable in the event of confronting an enemy. It enables an environment where an adversary is trapped into taking actions that consume/wastes his resources [1-3]. Employing deception and decoy techniques in network systems help to detect, trace, monitor and deter the activities of an adversary [4-6]. It is staged to make the adversary's life difficult where a false reality is projected as a reality to him. Indeed, Sir Sun Tzu encapsulated the art of deception in a perspicuous sentence when he said, "*The art of war teaches us not on the likelihood of the enemy's not coming, but on our own readiness to receive him; not on the chance of his not attacking, but rather on the fact that we have made our position unassailable*," [7].

We trace historical examples of the use of decoys to 1943 when the British found the corpse of a homeless man and went through extraordinary length to fabricate his death and created a prior but fake existing personal life for him to deceive the Germans. His realistic but fake persona included him been a captain in the military, having a father whom he sends/receives letters from and a fiancee's letter and photo. Also, the British planted some fake papers on him indicating a false location for an Allied attack. Of course, the Germans found the dead man's body and the letters on him. They read the letters and believed everything on it based on the '*supposed*' evidence found on him. Subsequently, they diverted their attention and military warfare to some other region. Unknown to them, they were conned and the Allied troop landed. For a long time, the German military continued to think that there was a diversion even after the Allied troop landed. History has it that this British facade of the human decoy saved over forty (40) thousand Allied lives [8].

In recent times, decoy/honey systems such as honeypots, honey tokens, honey accounts, honeywords which portray false resource have been deployed in various network systems to detect, observe

and thwart attacks from cybercriminals [9-10]. Honey encryption (HE) proposed by Juels and Ristenpart [11-12] is structured under the decoy-framework. It is an encryption scheme that yields valid-looking but fake message upon decryption with a wrong key thus an adversary gains no information about the original message.

The advent of highspeed and supercharged parallel and distributed systems (such as Graphical Processing Unit, Field Programmable Gate Arrays) paved the way for gathering, analyzing and processing large chunks of data often referred to as big data [13]. However, this huge advancement put cryptosystems at a disadvantage as attackers leverage on the high computational power of this systems to carry out brute-force attacks [14]. A predominant network attack that often jeopardizes computers connected to a network is the brute force attack. Even though conventional encryption schemes continue to guarantee security by increasing the size of the key or computational infeasibility of searching for the key, most cryptosystems fail to withstand cryptanalysis attack specifically the brute-force attack [15].

Honey encryption was proposed as a countermeasure to brute-force attack on conventional encryption schemes specifically for min-entropy systems like passwords [11-12]. It was observed from studies of persistent data breach that users chose simple, weak and predictable passwords which makes them susceptible to brute force attack [16-17]. Honey encryption acts as a supplementary encryption to fortify the conventional Password-Based Encryption (PBE) scheme.

Honey encryption addresses the flaws of password-based encryption schemes and is currently employed in securing most password-based system in the form of honeywords. However, honey encryption has not been employed in most systems such as, its application for encrypting human written documents like e-mails, etc. The challenge is how do we create contextually and semantically correct decoy-message that can actually fool an attacker? Other problems like typo-safety have remained unaddressed. For instance, if a legitimate receiver mistakenly enters a wrong password. Given that decoy system can completely address brute force attack which standard encryption schemes are susceptible to, then there is need to foster research in this area. Moreover, the current advance made on quantum computers propels us to search for quantum-safe cryptosystems. Since all our encryption schemes are exclusively based on Mathematical problems which are established based on the difficulty of solving discrete logarithm and number factorization problem [18], this puts us at an 'encryptionless' state as soon as quantum computers solve all the underlying Mathematics used to secure our modern cryptosystems. The HE scheme is not devised under the computational difficulty of breaking them alone but the real trickery that cryptography was meant to be built upon [10-12]. Consequently, honey encryption if properly designed and implemented, will be a good supplementary encryption scheme for the quantum era. Therefore, the target of this paper is to capture and present a synopsis of the current state of Honey encryption scheme to identify the gaps in the scheme to enable its optimization for real-life applications.

Our paper is organized as follows. First, we set the stage by presenting a detailed background of honey encryption scheme. Also, we present an up-to-date review of the literature in HE. Furthermore, we discuss the criteria of honey encryption, issues and challenges. We conclude with our propositions, suggestions and identification of promising areas for future research.

2. BACKGROUND OF RESEARCH

We give a brief overview of the conventional encryption to ensure a basic understanding of how Honey encryption evolved. In a conventional password-based encryption, an adversary performing a brute-force attack to obtain the key used for encrypting a message gets gibberish (non-uniform distribution) or an error symbol as the expected output when he tries a wrong key. This output is a pointer and distinguisher that the key he is trying is incorrect and he continues his search till he gets a plausible-looking message which may be the plaintext. During his attack, he quickly discards the message when the distribution is non-uniform. This gives him more time to continue his search, his probability of recovering the message/plaintext is high. Figure 1 shows a detailed explanation of how a conventional encryption scheme specifically Password-based encryption responds to a brute force attack.

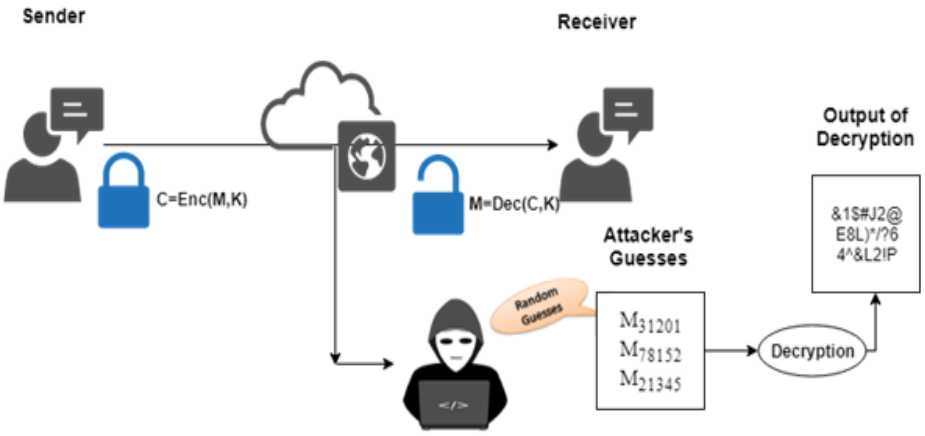

Figure 1. An illustration of the setting of the conventional encryption scheme in a brute-force attack

The sender encrypts his message using a key and a cipher. He sends the ciphertext to the receiver. The receiver decrypts the ciphertext using a decryption algorithm and the same key used by the sender. An attacker that intercepts the ciphertext may try to recover the message by randomly guessing the key. In the conventional setting scenario, the attacker can immediately tell from the suppose plaintext that the key he supplied is incorrect because of the non-uniform distribution of the plaintext. We describe an attack model of the HE scheme briefly.

For an encryption C=enc(M, K) of message M. If K and M are drawn from a known distribution. The target of an adversary is to recover the message M. He tries to decode C using different keys. For every key he tries, he gets M1,..., Mn. For a minimum entropy distribution like passwords, M is guaranteed to appear on his list. This is possible because users choose simple passwords that can be easily guessed. Also, attackers are aware of how users choose their passwords (from previously released details of leaked passwords on the internet). Therefore, the security here depends on the probability of the adversary been able to pick the message M from all n possible messages should one of the keys he tried was correct. In the event that an adversary correctly guesses the key, he is still stuck with a spoof data and cannot ascertain which is the correct message especially when he has no idea of the target message. He wins only if he can determine the message from the list of messages he recovered during his attack. Figure 2 shows a detailed explanation of how honey encryption scheme responds to a brute force attack.

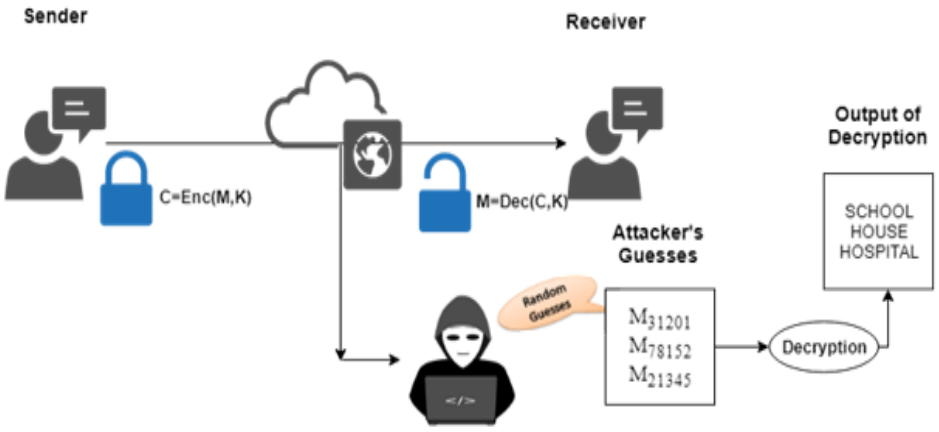

Figure 2. An illustration of the setting of the Honey encryption scheme in a brute-force attack

Distribution Transforming Encoder: Honey encryption works with a component referred to as the distribution transforming encoder (DTE). A DTE is a pair of algorithm DTE= (encode, decode) that takes M as an input and returns a value in S as output. Decode takes as input a value S and returns an output message

M. Honey encryption involves a DTE-and-then-encrypt process. This means a sender applies the DTE to the original message he intends encoding and then uses any conventional encryption scheme as the second layer of encryption. The DTE models all possible message relative to the original message and maps them to a seed space such that any key supplied when decrypting a message produces a relative, but fake message from the original message and this makes it difficult for the adversary to determine if he has recovered the original message or not. The DTE represents the model of the message. A good DTE is designed to model the message distribution well such that if a seed is selected uniformly at random and applied to it, the message is recovered. Figure 3a and 3b give a description of the DTE for encoding and its reverses, DTD for decoding a message.

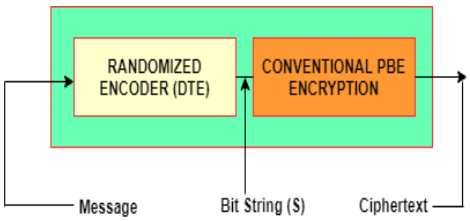

Figure 3a. Framework of DTE for Honey Encoding

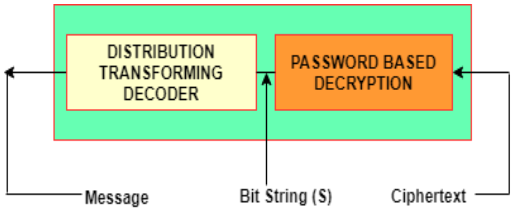

Figure 3b. Framework of DTD for Honey Decoding

The algorithm of the encode and decode process of HE is as follows [12]:

HEnc (K, M)	HDec (K, (R, C))
S ¬$ encode(M) R ¬$ {0, 1}^{n} S' ¬ H (R, K) C ¬ S'⊕ S return (R, C)	S'¬ H (R, K) S ¬ C ⊕ S' M ¬ decode(S) return M

H represents the cryptographic hash function, K represents a key, M represents the message, S represents the seed, R represents a random string, C represents the ciphertext, and ¬$ represents uniform random assignment.

2.1. Criteria for a Good Encoder

In order to create a plausible/convincing decoy that does not expose the original message to the adversary, it is fundamental to keep in mind two criteria:

a) Indistinguishability: Decoy messages must be difficult to distinguish from real messages. Successful deception of an adversary lies on the honey (decoy) message. Automated tools and humans should not be able to tell decoy message from true message. Decoys must be drawn from a probability distribution over possible messages similar to the distribution of the plaintext. Modeling human language requires honey messages that appear as they are used in real world. A good DTE must produce convincing decoy messages. The probability of telling decoy from true message must be minimal to successfully deceive the attacker.

b) Confidentiality: A good DTE must model the human language and at the same time hide the structural information of the original text. Human language such as e-mails, human-generated documents requires a considerable context-and-content relevant information. An encoder that does not reveal the structural skeleton of the underlying message/plaintext provides better security.

3. TOOLS FOR BUILDING ENCODERS

Several statistical tools employed for constructing the encoders to model natural language are discussed in this section.

3.1. Probabilistic Language Models

The language model is the probability distribution over sequences of words and it is used to predict the next word or next sentence. In this model, a trained n-gram generation model provides plausible letter sequences which will look very similar to words. Key in this case should set up the initial state of the sequence which would lead to a different combination of the letters. It is used to assign a probability to a sentence. Given a series of words K1, K2, K3.....Kt. It assigns a likelihood;

$$P(K) = P(K1, K2, K3, \dots, Kt) \quad (1)$$

3.2. Markov Models

A stochastic model used to represent randomly changing systems where future states depend on current states and do not depend on past events that occurred before it. Predictable events that occur over time are fashioned using Markov models. Markov model is the Markov chain which models the state of a system with a stochastic variable that changes over time.

3.3. Grammar Model or Code-book Method

Generation method where syntactical trees and dictionaries are used to generate plausible text or the code-book where an existing book with text and intervals is used to encode the combination of the words.

4. PROPOSED METHODS TO IMPROVE THE HONEY ENCRYPTION SCHEME

This section describes proposals from several researchers in the past. The proposal section is divided into two groups. The first part is focused on the encoder of the DTE. The second part describes proposals on typo problems in HE. Table 1 depicts a description of models given by different authors.

4.1. Typo-Problems in Honey encryption

In the HE scheme, a user is supplied with a seemingly real but false text for every key he supplies. A significant drawback with the HE scheme is typo safety when a legitimate user mistakenly enters an incorrect key. Typo error is a serious problem in HE if not adequately provided for. This section provides a review of research proposed under typo security in honey encryption scheme. Table 2 depicts a description of typo-based security solution provided by different authors.

5. ISSUES AND CHALLENGES

As discussed in the introduction section, HE provides security beyond brute-force bound; in settings where minimum entropy keys are used to secure messages. However, there are open areas in HE which need to be researched. In this section, we briefly describe some of the problems of HE.

a. Honey encryption is difficult to apply in a setting where the plaintext is large or the distribution of the message is unknown. A large plaintext requires a substantial content to be used to construct the DTE so that fake-texts that looks like the original texts can be used as the decoy message. The contents of the decoy message also need to have a good contextual meaning relative to the original message. The authors acknowledged this difficulty in their paper when they pointed out that *"...The key challenges of honey encryption scheme are development of appropriate instances of a new type of randomized message encoding scheme called a distribution-transforming encoder (DTE)"* [11-13]. This problem was also strengthened when Juels [9] pointed out the complexity of encoding honey documents, for instance, e-mails require generating fake but semantically and contextually realistic natural language message.
b. Having a context-sensitive decoy produces good instances of the message but if not correctly constructed will reveal the structure of the original message giving the adversary an upper hand/high probability of recovering the message using chosen-ciphertext attack (CCA).
c. HE is tailored to work in low-entropy settings like passwords, RSA keys, PINS and Credit card numbers. Extending it to support other settings and file type requires an extensive design of the DTE to meet the criteria discussed in section 2.1.
d. HE is fashioned to produce fake but valid-looking text for every key supplied by anyone. Therefore, a legitimate user that made a typo error while trying to retrieve an encrypted message will recover a valid-looking but fake message and he has no way of knowing this.

Table 1. Description of models given by different authors

Authors	Description	Justification	Flaws
Jo et al. [19]	❖ The authors proposed the statistical code scheme which is a unification of the structural code scheme and honey encryption scheme. In this proposal, the HE provides the semantic feature of language while the structural code scheme provides the syntactic features of natural language. ❖ The performance was evaluated to find how many times it was required to generate meaningful false text from the original text in the corpus. The probability was 0.38 which means at least four repetitions are required to generate convincing decoy message.	❖ This approach provides plausible false text relative to the original text as decoys.	❖ It does not support other data-format but generates decoys only for short length message. ❖ The ambiguity between the original plaintext and the false text is much, and the adversary may use this to figure out the difference between the false text and original text when he has a little knowledge of the target message
Chatterjee et al. [20]	❖ This proposal describes how to build a Natural Language Encoder (NLE) called NoCrack using existing password models. ❖ The performance was evaluated by measuring the time required to recover a particular vault and the time to add a password to a vault. ❖ The analysis showed that, the smaller the vault, the faster the recovery. Large vaults require a long time to generate a decoy vault.	❖ This proposal performs best when the vault is small. ❖ A single password can be recovered quickly. ❖ This approach creates realistic decoy vaults on the fly during brute-force attacks.	❖ This approach does not support large vault as it is very slow. ❖ The system reflects human language but was borne in the context of password security and does not support human-written documents like emails or long messages.
Golla et al. [21]	❖ This proposal used Kullback–Leibler (KL) divergence to prove that the approach by [20] degrades security. ❖ The proposal also pointed out that all fixed NLE is susceptible to the KL divergence attack and proposed the adaptive NLE. ❖ The adaptive NLE was constructed using Markov model. It was evaluated using KL divergence attack, and the analysis showed that the real vault ranks among 40.12% of the most likely vault.	❖ This proposal improves previous methods of securing the vaults for password security. ❖ The adaptive NLE increases the message space, allowing more instances of online guessing of the original vault.	❖ One of the major drawbacks of adopting the fixed or static NLE is intersection attack. ❖ There is no adequate closure of providing maximal security.
Beunardeau et al. [22]	❖ This proposal contends that the proposed method by [20-21] works reasonably well to secure short passwords but fail to model natural language as used in real-world scenarios such as e-mails and written documents. ❖ This proposal explains that context-relevant information is required to model human language to produce convincing decoy messages that fool human and automated tools. ❖ The corpus Quotation DTE is proposed.. Grammar model of language is used to build the DTE and users are required to only quote from a known public document.	❖ This proposal suggests how to extend the scheme to allow encoding human-written texts.	❖ Quoting from a public document restrict users to the vocabulary of the document domain. ❖ Fixed codebook is not able to provide all the required combinations of words. For instance, it is unlikely that a user will be able to encode a text from a computer science domain using a code-book from a flower-based domain.

Table 2. Description of typo-based solution given by different authors

Authors	Description	Positives and Negatives
Chatterjee et al. [23]	❖ This proposal presents a typo-tolerant checker which works relatively well with the existing password authentication system. ❖ This proposal pointed out that at least a minute would have been saved for 20% of the users if the typo-tolerant scheme is adopted.	This approach is suitable for existing password-based authentication system as it applies caps lock corrector, first case flip corrector and also extra character at the end corrector to improve usability but this approach is not suitable to handle the typo problem in the HE scheme.
Choi et al. [24]	❖ The scheme provided two types of typo-safety both in an offline and online setting to handle different typo problems while still retaining message recovery in a typical HE scheme.	The Type A protocol is easier to implement as it requires only a server but the major drawback is that the size of the key is small and also there is the uncertainty of detecting typos in some settings. Type B is an improvement over Type A as a user can easily notice typos if he remembers his pin, however, a key problem here is that user has to remember the pin to verify his message.
Chatterjee et al. [25]	❖ This proposal presents a personalized typo tolerant password checking. ❖ This research proposes a simple blacklisting procedure in which a small set of risky typos is prohibited from being admissible into the typo cache.	This research is an improvement over existing typo-tolerant password schemes but is not designed to work on typos committed on a decoy system. However, it can be modified for Honey encryption.

6. CONCLUSIONS AND FUTURE CHALLENGES

In this paper, issues, challenges and detailed literature review of the Honey Encryption (HE) scheme is provided. The aim is to furnish current/aspiring researchers and practitioners with a comprehensive overview of the state-of-art research in the scheme. From the survey of various proposals, we conclude that the current techniques used in producing decoy message do not model human language entirely and so fail to produce decoys that are acceptable and convincing to lure the attacker away from the genuine resource.

Presently, honey encryption scheme has been implemented for credit card numbers, passwords and RSA pins. There is an urgent need for HE to be adapted for other settings such as decoys for human generated message such as e-mails, convincing decoys to confront eavesdropping attack during online chatting, etc.

This study has raised many questions in need of further investigation. Therefore, we propose further research in the following areas:

a. Natural Language Processing in Honey Encryption: How do we capture the empirical properties of language? How do we model the human language itself as an effective tool for designing convincing decoys?
b. How do we honey encrypt (produce decoys) without revealing the structure of the message?
c. How do we generate decoy/honey messages that fool machines and human from realizing real messages from decoy messages?
d. How do we handle mauling and prevent adversaries from learning partial information of the original message from the decoy during an exhaustive key-search?
e. How do we address typo problems in the H.E scheme? This problem requires immediate attention and extensive research even before any implementation of the HE scheme.

REFERENCES

[1] Omolara AE, Jantan A, Abiodun OI, Poston HE. A Novel Approach for the Adaptation of Honey Encryption to Support Natural Language Message. In *Proceedings of the International MultiConference of Engineers and Computer Scientists* 2018 (Vol. 1).
[2] Disso JP, Jones K, Bailey S. A plausible solution to SCADA security honeypot systems. In *Broadband and Wireless Computing, Communication and Applications* (BWCCA), 2013 Eighth International Conference on 2013 Oct 28 (pp. 443-448). IEEE.
[3] Bringer ML, Chelmecki CA, Fujinoki H. A survey: Recent advances and future trends in honeypot research. *International Journal of Computer Network and Information Security*. 2012 Sep 1;4(10):63.
[4] Dhanji PK, Singh SK. Assault Discovery and Localizing Adversary in Remote Networks. *Indonesian Journal of Electrical Engineering and Computer Science*. 2018 Jan 1;9(1): 81-84
[5] Zhang D. Inconsistency: the good, the bad, and the ugly. In *Information Reuse & Integration*, 2009. IRI'09. IEEE International Conference on 2009 Aug 10 (pp. 182-187). IEEE.
[6] Pawlick J, Colbert E, Zhu Q. A Game-Theoretic Taxonomy and Survey of Defensive Deception for Cybersecurity and Privacy. *arXiv* preprint arXiv:1712.05441. 2017 Dec 14.

[7] S. Tzu. "The art of war." In *Strategic Studies*, pp. 63-91. Routledge, 2008.
[8] Montagu, E. The Man Who Never Was, J. B. Lippincott Company, Philadelphia, PA (1954).
[9] Juels A. A bodyguard of lies: the use of honey objects in information security. In *Proceedings of the 19th ACM symposium on Access control models and technologies* 2014 Jun 25 (pp. 1-4). ACM.
[10] Juels A, Rivest RL. *Honeywords: Making password-cracking detectable*. In Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security 2013 Nov 4 (pp. 145-160). ACM.
[11] Juels A, Ristenpart T. Honey Encryption: Encryption beyond the brute-force barrier. *IEEE Security & Privacy*. 2014 Jul;12(4):59-62.
[12] Juels A, Ristenpart T. Honey encryption: Security beyond the brute-force bound. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques* 2014 May 11 (pp. 293-310). Springer, Berlin, Heidelberg.
[13] Kwok SH, Lam EY. Effective uses of FPGAs for brute-force attack on rc4 ciphers. *IEEE Transactions on Very Large-Scale Integration (VLSI) Systems*. 2008 Aug;16(8):1096-100.
[14] Couture N, Kent KB. The effectiveness of brute force attacks on RC4. In *Communication Networks and Services Research*, 2004. Proceedings. Second Annual Conference on 2004 May 19 (pp. 333-336). IEEE.
[15] Najafabadi MM, Khoshgoftaar TM, Kemp C, Seliya N, Zuech R. Machine learning for detecting brute force attacks at the network level. In *Bioinformatics and Bioengineering* (BIBE), 2014 IEEE International Conference on 2014 Nov 10 (pp. 379-385). IEEE.
[16] Bonneau J. The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *Security and Privacy* (SP), 2012 IEEE Symposium on 2012 May 20 (pp. 538-552). IEEE.
[17] Florencio D, Herley C. A large-scale study of web password habits. In *Proceedings of the 16th international conference on World Wide Web* 2007 May 8 (pp. 657-666). ACM.
[18] Gamido HV, Sison AM, Medina RP. Modified AES for Text and Image Encryption. *Indonesian Journal of Electrical Engineering and Computer Science*. 2018 May 27;11(3).
[19] Jo HJ, Yoon JW. A new countermeasure against brute-force attacks that use high performance computers for big data analysis. *International Journal of Distributed Sensor Networks*. 2015 Jun 1;11(6):406915.
[20] Chatterjee R, Bonneau J, Juels A, Ristenpart T. Cracking-resistant password vaults using natural language encoders. In *Security and Privacy* (SP), 2015 IEEE Symposium on 2015 May 17 (pp. 481-498). IEEE.
[21] Golla M, Beuscher B, Dürmuth M. On the security of cracking-resistant password vaults. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* 2016 Oct 24 (pp. 1230-1241). ACM.
[22] Beunardeau M, Ferradi H, Géraud R, Naccache D. Honey Encryption for Language. In *International Conference on Cryptology in Malaysia* 2016 Dec 1 (pp. 127-144). Springer, Cham.
[23] Chatterjee R, Athayle A, Akhawe D, Juels A, Ristenpart T. pASSWORD tYPOS and how to correct them securely. In *Security and Privacy* (SP), 2016 IEEE Symposium on 2016 May 22 (pp. 799-818). IEEE.
[24] Choi H, Nam H, Hur J. Password typos resilience in honey encryption. In *Information Networking (ICOIN)*, 2017 International Conference on 2017 Jan 11 (pp. 593-598). IEEE.
[25] Chatterjee R, Woodage J, Pnueli Y, Chowdhury A, Ristenpart T. *The TypTop System: Personalized Typo-tolerant Password Checking*. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security 2017: 329-346