JOIN (Jurnal Online Informatika)
p-ISSN: 2528-1682, e-ISSN: 2527-9165
Volume 11 Number 2 | December 2026: 256-268
DOI: 10.15575/join.v11i2.1776

Increasing Plaintext Indistinguishability in Honey Encryption using Context Relation-based Decoy Chat Message

Yusril Firza[1], Ari Moesriami Barmawi[2]
[1,2]Magister of Cyber Security and Digital Forensics, Telkom University Bandung, Indonesia

Article Info

Article history:

Received July 26, 2025
Revised September 5, 2025
Accepted December 5, 2025
Published September 17, 2026

Keywords:

Context Based
Decoy Message
Electra
Honey Encryption
KeyBERT

ABSTRACT

General encryption method provides a low level of security against brute-force attacks, which attempt every possible key through the decryption process. The Honey Encryption (HE) method was developed to overcome these problems. However, the latest development of HE implemented in chat conversations results in conversations that do not correlate between one chat and another when the ciphertext is decrypted using the wrong key (decoy message). The absence of context relation in the decoy message results in the decryption results of the brute force attack becoming easy to analyze, which can make it possible for the adversary to obtain the correct decryption results and map the keys used. This study aims to improve the context relation or contextual naturalness between chats in HE decoy messages. This improvement is intended to generate contextually natural sentences such that an attacker cannot distinguish between messages decrypted using the correct key and those decrypted using the incorrect key. This process reduces the likelihood of success during a brute-force attack for obtaining the plaintext and the key.. A contextual naturalness decoy message is generated by utilizing keyword detection (KeyBERT) and keyword alternative mapping as key points for context generation. Additionally, the generated keywords are used in sentence generation with a Masked Language Model (in this study, Electra) for generating word candidates. The implementation of KeyBERT and Electra in this study shows that 22 out of 32 decoy chat messages can be categorized as natural conversations. Finally, it can be concluded that the proposed method has contextual relation between decoy chat messages.

Corresponding Author:

Ari Moesriami Barmawi,
Magister of Cyber Security and Digital Forensics, Faculty of Informatic, Telkom University Bandung
Jl. Telekomunikasi No. 1, Bandung Terusan Buahbatu - Bojongsoang, Sukapura, Kec. Dayeuhkolot, Kabupaten Bandung, Jawa Barat
Email: aribarmawi@telkomuniversity.ac.id

1. INTRODUCTION

Encryption methods in general are still vulnerable to key guessing using brute force attacks. This vulnerability is based on decryption results that can be analyzed to determine whether the used key is correct or not. Using Honey Encryption (HE) as an encryption method can increase security against existing boundaries in the face of brute-force attacks [1]. By implementing a distribution-transforming encoder (DTE) as a new type of message randomization coding scheme in password-based encryption methods, HE can map the decryption result so that the decryption process always yields a plausible result to be considered as the original plaintext. However, HE has limitations, such as the

limited number of plaintexts that can be accommodated. In its first development, HE was limited to credit card numbers, PINs, CVVs, and RSA secret keys [1].

Omolara's research overcame the limitations of HE by implementing natural language processing. In this method, the decoy message generation utilizes a recurrent neural network (RNN)[2] and the Term Frequency-Inverse Document Frequency (TF-IDF) approach to extract keywords and form sentences [3]. During encryption, the method modifies the keyword set using hypernyms and hyponyms from WordNet [4]. This set of keywords is then used as a parameter for generating sentences during the decryption process. HE in Omolara's method can create fake messages when the ciphertext is decrypted using an incorrect key. This fake message is called a decoy message. Omolara's method has the decoy messages with less than 50% of the messages that can be guessed. However, this research only produces one-sentence decoy messages [5]. This limitation has been overcome by Esther Omolara Abiodun et al. [6] by implementing Natural Language Encoding, allowing sentences in plaintext to be used in HE. The development of instant messenger has been tested as a case study. Although it yields good results, such as a few grammatical errors, the method generates decoy messages without any contextual connection between chats. This condition occurred because the research focused on sentence processing.

In this research, development is carried out regarding the absence of contextual connection between chats (contextual naturalness) when the decryption process in HE is carried out with an incorrect key. To provide context between chats, this research focuses on utilizing keywords and an appropriate list of word candidates scheme as a seed table. This research examines the contextual naturalness from human and machine perspectives. Contextual naturalness is assessed through structured sentence writing and the continuity of context in chats. Based on the experiment's results in its scenario, these results can improve the level of contextual naturalness in decoy chat messages generated by HE from both human and machine perspectives.

In this study, decoy messages were generated by combining KeyBERT for keyword detection. For modifying the seed table generation, the Electra Masked Language Model is used for generating a list of words in the seed table. The implementation of KeyBERT and Electra was necessary to ensure that the decoy messages had a controlled context and a natural-sounding chat. Natural decoy messages and controlled conversation contexts can reduce adversaries' suspicion regarding the authenticity of the generated decoy messages during brute-force attacks.

2. METHOD

This proposed method aims to overcome the problem of the Omolara method, which is the absence of context relations between decoy chat messages. To overcome the problem, keyword utilization and seed table generation are proposed. For implementation, KeyBERT is used for keyword detection, and the Electra Masked Language Model is used for seed table generation. The main processes of the proposed encryption method are keyword processing, seed table generation, and encryption. Keyword processing in encryption utilizes three sub-processes, which are keyword detection using KeyBERT, alternative keyword mapping, and identifier generation. Furthermore, the seed table generation includes three processes to generate different types of seed tables, "Yes No Question" seed tables, pronoun seed tables, and general seed tables that utilize Electra. Finally, the chat is encrypted using Honey Encryption and AES. The output of the encryption is the identifier and the ciphertext. The decryption process includes keyword processing, seed table generation, and decryption. Keyword processing consists of keyword generation obtained from the identifier. Seed table generation has the same process as the seed table generation in the encryption process. After the seed table is generated, decryption is performed using AES and Honey Encryption. Honey encryption mapped the decryption result using the Distribution Transforming Encoder (DTE) to make the decryption output always readable. The output of the decryption process is the chat message. By implementing Honey Encryption, the decryption results are decoy messages when the ciphertext is decrypted using the incorrect keys. Otherwise, the decryption result is the original message.

2.1. Encryption Process

This method implements three major processes in the encryption process (see Fig. 1). The first is keyword processing. It detects keywords in plaintext sentences, maps alternative keywords, and processes identifiers as parameters for generating the seed table during the decryption process. The second process is seed table generation. This process generates all the necessary seed tables, including

p-ISSN: 2528-1682
e-ISSN: 2527-9165

the Yes/No Question, Pronoun, and General seed tables. The last process is encryption itself. The distribution-transforming encoder processes each word before the entire seed is encrypted using AES [7].

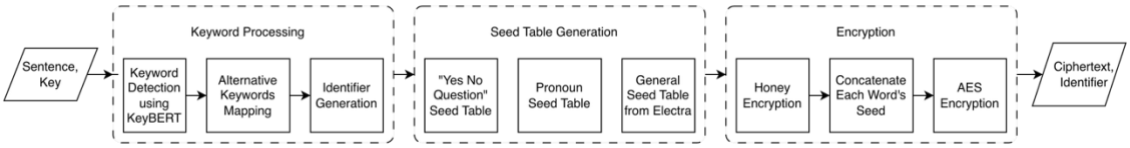

Figure 1. General Method of Encryption

2.1.1. Keyword Processing

This process is aimed to produce keyword information as the key point of word candidates' (seed tables) generation. Keyword processing consists of three processes: keyword detection, alternative keywords mapping, and identifier generation. The output includes the keyword of the processed sentence, alternative keywords that add context to the decoy message, the keyword ID, and the keyword code. Additionally, there is an identifier creation process. This identifier is sent with the ciphertext such that the decryption process uses the same seed table as the encryption process. The detail of the processes are as follows:

1. Keyword Detection using KeyBERT

In this process, a keyword is used to generate the word representation of the sentence. To obtain the sentence keywords, this proposed method utilizes KeyBERT, which performs feature extraction by calculating the vector for each word in the sentence being processed (see Fig. 2).

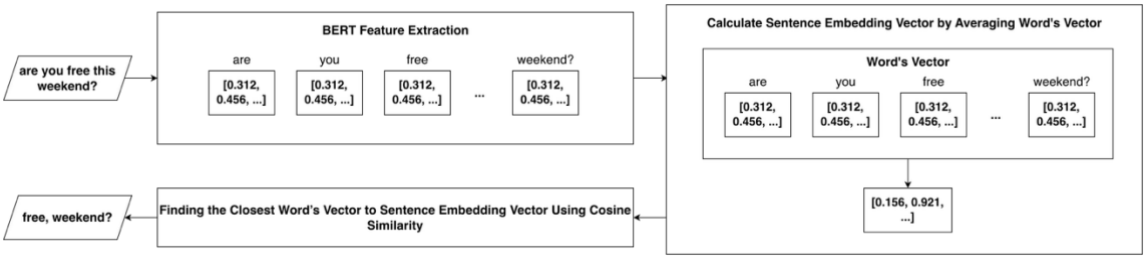

Figure 2. Keyword Detection Simulation of KeyBERT

KeyBERT is a method used to extract keywords from a sentence. This method utilizes feature extraction in Scikit-Learn to obtain vectors from each word in the processed sentence. Then, KeyBERT calculates the sentence vector by calculating the average value of the vectors that have been obtained. This method extracts keywords based on the closest distance between word vectors in a sentence and sentence vectors calculated using Cosine Similarity from word vector A to sentence vector B (see Equation 1).

$$cos = \frac{A \cdot B}{||A||\,||B||} \quad 1)$$

Feature extraction in KeyBERT utilizes CountVector in Scikit-Learn to tokenize the vector for each word. After that, each vector in each word (w_i={ $w_{i\_0}, w_{i\_1}, \dots, w_{i\_n}$}) is averaged [8] using Equation 2 to get the sentence embedding vector (W) using Equation 3. After the entire vector is obtained, the distance of each vector to its sentence embedding vector is calculated using cosine similarity (see Equation 4). Based on the distance vector of each word, the words with the closest distance to the sentence embedding are labeled as keywords [9].

$$Avg_{vec} = \frac{1}{n}\sum_{i=0}^{n} vec_i$$

$$W = [Avg_{vec\_0}, Avg_{vec\_1}, \dots, Avg_{vec\_n}]$$

$$Sim_i = cos(w_i, W)$$

2. Alternative Keyword Mapping

This process is aimed to create various decoy message contexts related to the given decryption key. Alternative keyword mapping involves creating a set of keywords used as context keywords in decoy messages. These alternative keywords are obtained by mapping original keywords to possible replacement words in the same position in the sentence for each context possibility.

3. Identifier Generation

This process aims to generate the information needed for the decryption process to generate the same seed table as the one generated during encryption. This process ensures that the encryption and decryption processes produce aligned outputs. The components of the identifier consist of several information, including those described below.

A. Pronoun Identifier

A pronoun identifier is used to make the decoy message have a relevant pronoun as the sentence subject in the context of a conversation. Pronoun identifiers are generated when the system processes a word that indicates a pronoun. This process considers the pronoun identifier from the previous chat. The pronoun identifier contains information about the sentence being processed, including consistency, pronoun ID, and pronoun type.

a. Consistency. Pronoun consistency is defined as the similarity between pronouns in the current chat and those in the previous chat. If the pronoun is the same as the one in the previous chat, the consistency score is 1 (true). Otherwise, the consistency score is 0 (false).
b. Pronoun ID. A pronoun ID marker indicates the presence of a pronoun in a given sentence. This data is necessary for processing the pronoun identifier in the next sentence.
c. Pronoun Type. There are two types of pronouns: personal and possessive. Personal pronouns represent people, while possessive pronouns represent ownership.

B. Keyword Identifier

The proposed method also requires some information about the keyword being processed during decryption. Therefore, this information needs to be sent through the identifier. This process is conducted to maintain the consistency of the seed table created by the system during the decryption process. Some attributes of the keywords sent through the identifier are shown as follows.

a. Keyword ID

To obtain decryption results that have various conversation contexts, it is necessary to encode keywords in each sentence. For this purpose, the keyword ID is taken based on the keyword ID in the keyword list that has already been mapped in the alternative keyword mapping. Furthermore, the keyword ID is encrypted using HE with AES as the encryption method. In Equation 5, the set of keywords treated as seeds (S) is encrypted using AES and the same key (K) as used in the plaintext encryption.

$$keyword_{id} = AESEnc(S, K) \quad (5)$$

p-ISSN: 2528-1682
e-ISSN: 2527-9165

b. Keyword Code

The keyword code is used to identify whether a word is a keyword or not. Each word in the sentence has its code. If the words in the sentence are not keywords, then the code of the word is 0. If the words in the sentence are keywords, then the code of the word is 1. If we have "are you free this weekend?" as the sentence and the keywords are "free" and "weekend?", the keyword code is 00101 as shown in Fig. 3.

Sentence	are	you	free	this	weekend?
Keyword			free		weekend?
Keyword Code	0	0	1	0	1

Figure 3. Keyword Code Illustration

C. "Yes-No Question" Status

"Yes-No Question" status is used to identify whether a sentence is a Yes-No question or not. The data for this attribute is obtained by reading the first word in the processed sentence. Then, the word is checked based on the "Yes-No Question" seed table. If the first word is in the seed table (First Word Sentence ∈ Seed Table), then the value of this parameter is 1. Otherwise, the value of this attribute is 0.

2.1.2. Seed Table Generation

The seed table contains words that serve as candidates to replace non-keywords in processed sentences. These words are obtained through keyword mapping. Any word that is not a keyword can be replaced during decryption and has its seed table based on its type. There are three necessary seed tables, which are the Pronoun Seed Table, the Yes-No Question Seed Table, and the General Seed Table. The definitions of these three seed tables are as follows:

1. Pronoun Seed Table

   This seed table is created to prevent the system from generating sentences that are out of the context of the conversation. In this seed table, the seed table is divided into two types. Seed table for personal pronouns and seed table for possessive pronouns. In this seed table, there are not only the list of pronoun words but also their word pair (such as "you" and "I"). This seed table is done to accommodate two-way conversations that have interchangeable pronouns.
2. Yes-No Question Seed Table

   This seed table is used to ensure the decryption result has the "Yes-No Question" pattern when the "Yes-No Question" status is true. The seed table for the sentence "Yes-No Question" is created by calling the list of possible first words in the sentence "Yes-No Question". This list has been mapped based on the existing "Yes-No Question" sentence set. Each word in this seed table has its ID, so that each first word in the "Yes-No Question" being processed can have its seed based on the ID of the word.
3. General Seed Table from Electra

   This seed table is created using the Masked Language Model (MLM) [10]. This method provides word recommendations to replace masked words in sentences. In this study, we used the MLM from Electra.

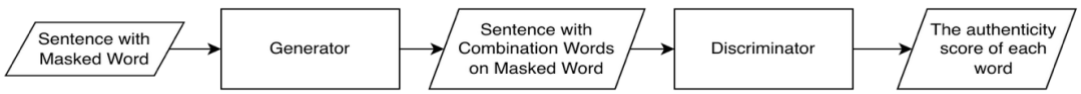

Figure 4. Electra Masked Language Model Process

Electra is used to perform a list of suitable words to complete sentences that have blank words by masking the words that are blank words. Then, the sentence is processed in Electra using a Generator and Discriminator (see Fig. 4). The Generator process utilizes an existing MLM model (usually a small MLM) to obtain a vocabulary list. This process generates a new sentence by replacing the previously masked word m (Equation 8, 10). Generator also provides probability score of token sequence x into sequence contextualized vector representations h for given position t through Equation 6.

The sentence is then processed by the Discriminator. This process calculates the loss value of the word tokens in the sentence (see Equation 10-14) to determine its originality value through Equation 9 for each mask m that utilize sigmoid (Equation 7) for each token weight vectors w [11].

$$P_G(x) = exp\ exp\ (e(x_t)^T h_G(x)_t) \Big/ \sum_{x'} exp(e(x')^T h_G(x)_t) \quad (6)$$

$$D(x,t) = sigmoid(w^T h_D(x)_t) \quad (7)$$

$$m_i \sim unif\{1,n\}\ for\ i = 1\ to\ k \quad (8)$$

$$\hat{x_i} \sim P_G(x^{masked})\ for\ i \in m \quad (9)$$

$$x^{masked} = REPLACE(x, m, [MASK]) \quad (10)$$

$$x^{corrupt} = REPLACE(x, m, \hat{x}) \quad (11)$$

$$L_{MLM}(x, \theta_G) = E\left(\sum_{i \in m} -logP_G(x_i | x^{masked})\right) \quad (12)$$

$$L_{Disc}(x, \theta_D) = E\left(\sum_{t=1}^{n} -I(x_t^{corrupt} = x_t)\ log\ log\ D(x^{corrupt}, t) - I(x_t^{corrupt} \neq x_t) log(1 - D(x^{corrupt}, t))\right) \quad (13)$$

$$min_{\theta_G, \theta_D} \sum_{x \in X} L_{MLM}(x, \theta_G) + \lambda L_{Disc}(x, \theta_D) \quad (14)$$

This method masked non-keywords with "[MASK]" as input to Electra. For example, if the plaintext is "Are you free this weekend?" and the keywords are "free" and "weekend," this method will insert "[MASK] [MASK] free [MASK] weekend?" as the input sentence for Electra (as shown in Fig. 5). The masked word is treated as an empty word that should be filled with one of the word candidates in the seed table. In Electra, as seen in Equation 6, each list of word candidates (x') is scored using $pG(x_t \mid X)$ in the context of sentence naturalness if the list of word candidates filled the empty word ($e(x_t)$) through the hidden state of the Electra generator ($h_G(x)_t$).

Sentence	are	you	free	this	weekend?
Keyword			free		weekend?
Keyword Code	0	0	1	0	1
Masked Sentence	[MASK]	[MASK]	free	[MASK]	weekend?

Figure 5. Masked Sentence Illustration

2.1.3. Encryption

The encryption process utilizes Honey Encryption and AES. In this study, Honey Encryption is used to ensure that the encryption produces ciphertext that can be processed by Honey Encryption so that a decoy message can be obtained when the decryption key is incorrect.

2.1.3.1. Honey Encryption

Honey Encryption (HE) is an encryption method that provides security beyond the limitations of brute-force attacks using Distribution Transforming Encoder (DTE) [12] . HE constructs a ciphertext

p-ISSN: 2528-1682
e-ISSN: 2527-9165

C that, when decrypted with any key (whether using the correct or incorrect key), produces a message that appears reasonable or natural. With Honey Encryption, an encrypted message M will create a fake message M' that appears equally valid to an attacker as M, when decrypted using an incorrect key K'. To achieve this, the HE schemes implements a specialized encoding mechanism and then encrypts the result with a conventional password-based encryption scheme. Suppose a message from Alice M is encrypted using HE and key K, then decrypted with an incorrect key K' to produce a fake message M'. An attacker attempting to implement this K' will believe they have successfully obtained the correct key because the resulting message appears reasonable or natural [13]. Thus, the probability of breaking the key becomes infinite, as the attacker cannot distinguish between the decryption results obtained using the incorrect key and those obtained using the correct key. The overview of the scheme, in the case when using incorrect and correct keys, is shown in Fig. 6, 7, and 8.

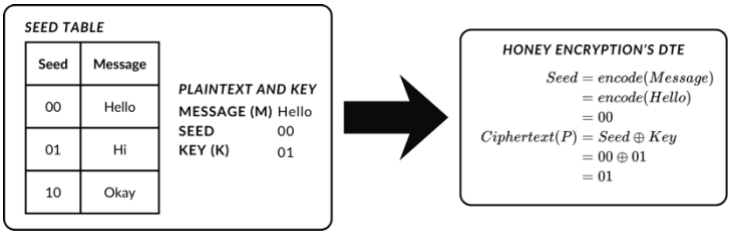

Figure 6. Honey Encryption Equation Simulation

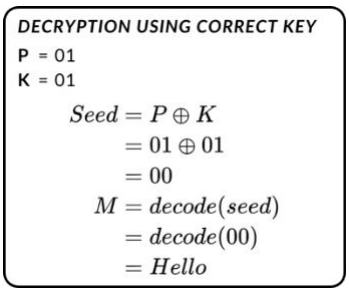

Figure 7. Honey Encryption Correct Key Decryption Simulation

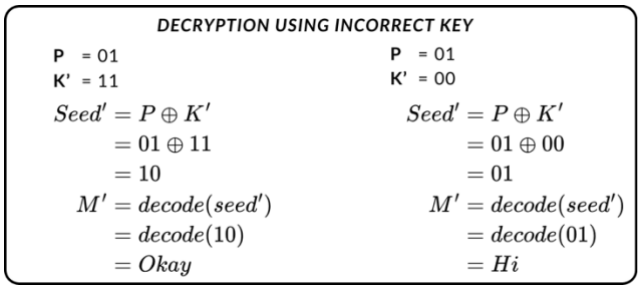

Figure 8. Honey Encryption Incorrect Key Decryption Simulation

The encryption process involves processing each word in the plaintext sentence. Each word has its seed table, except for the keyword of the sentence. The seed table and non-keyword are the input of the distribution-transforming encoder from Honey Encryption (see section 1.1). The encoder's output is the seed for each word. The seed of keywords is filled with zero by default. After obtaining the seeds, they are encrypted using an existing encryption method.

2.1.3.2. AES Encryption

This research experiment uses AES symmetric encryption, which is also used by the previous method [14]. The plaintext input for AES is the concatenated seed of each word. After that, the ciphertext and the identifier are sent together to ensure that the decryption process uses the same seed table for each word.

2.2. Decryption Process

In the decryption process, there are three processes. These processes are keyword generation, ciphertext decryption, and seed table generation. The input to the keyword generation and seed table generation processes is the information that consists in the identifier received by the system. The overview of the decryption process is shown in Fig. 9, and the example of the decryption result using the wrong key is shown in Table 1.

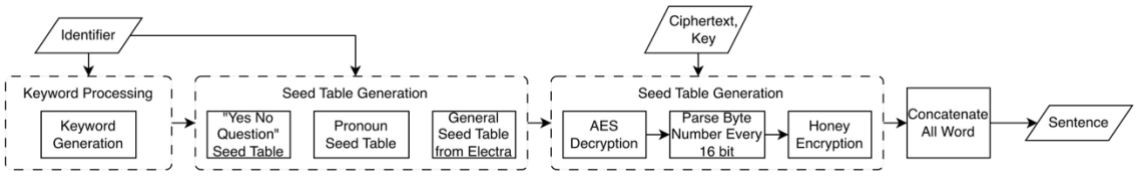

Figure 9. General Method of Decryption

Table 1. Decryption Result Example Using Correct and Incorrect Key

Plaintext	Correct Key	Incorrect Key
Are you free this weekend? Yeah	Are you free this weekend? Yeah	Should I know that song? Yeah
I was thinking about getting a motorcycle ride tomorrow.	I was thinking about getting a motorcycle ride tomorrow.	I am wanting to come to Adelle's concert tomorrow.
I've been wanting to go for a ride. let's meet at the usual spot	I've been wanting to go for a ride. let's meet at the usual spot	I've been wanting to be at a concert. Let's meet up at the concert venue.

2.2.1. Keyword Processing

Keyword generation is conducted to obtain keywords for sentence generation using the keyword ID. This process requires the encrypted Keyword ID from the identifier. This keyword is processed by implementing the HE distribution transforming decoder (see Equation 16) to create a different keyword ID when decryption is performed with an incorrect key. Equation 15 is used in the decryption of the encrypted keyword ID. The decryption result is used as input for the distribution transformation decoder that is formulated as shown in Equation 17 to obtain the seed as the keyword ID. The seed is decoded using a list of alternative keywords to determine keywords that will be processed in the future.

$$Keyword'_{seed} = AESDec(Keyword_{id}, Key) \quad (15)$$

$$Keyword_{seed} = Keyword'_{seed} \; mod \; (max\_alternative\_keyword\_ID + 1) \quad (16)$$

$$Keyword = Decode(Keyword_{seed}) \quad (17)$$

2.2.2. Seed Table Generation

In the decryption process, the process to generate the seed table is similar to that used in encryption. However, the use of those three seed tables is different from the encryption. Those three seed tables are used as follows:

1. “Yes-No Question” Seed Table. The "Yes/No Question" seed table contains the same information as the seed table used for encryption. The difference lies in how it is used. While the encryption side detects the “Yes-No Question” status of a sentence based on the first word, the decryption side is slightly different. On the decryption side, the "Yes/No Question" status in the identifier is observed if the yes-no question status shows a true value (or 1). In this case, the yes-no question seed table is used for the first word of the sentence being processed. Meanwhile, if the “Yes-No Question” status is 0, the seed table used for the first word is the general seed table.

p-ISSN: 2528-1682
e-ISSN: 2527-9165

2. General Seed Table from Electra. Similar to the encryption process, the general seed table uses Electra's MLM. The difference is that the system provides Electra with input obtained from the keyword code in the identifier and the keywords generated in the initial process. For example, if the system receives the keyword code 00101 and the keywords "free" and "weekend?" from the identifier, then the disguised sentence becomes "[MASK] [MASK] free [MASK] weekend?" (see Fig. 10).

Keyword Code	0	0	1	0	1
Keyword			free		weekend?
Masked Sentence	[MASK]	[MASK]	free	[MASK]	weekend?

Figure 10 Masked Sentence from Keyword Code

3. Pronoun Seed Table. The pronoun seed table is used to correct the pronouns obtained by the system. This process occurs after the system receives a word from the general seed table. The pronoun seed table is used to re-correct words when the system receives a pronoun from the general seed table. This process ensures that the decrypted sentence has the appropriate conversation pattern as shown in Fig. 11.

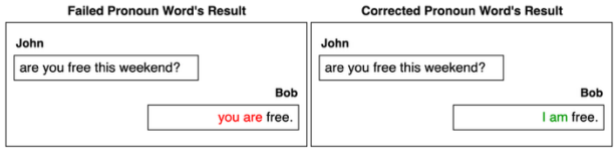

Figure 11 Example of Pronoun Word Correction

2.2.3. Decryption

Decryption involves processing the ciphertext using the same method that was used for encryption. After decryption, the system splits the result into 16-bit segments. Each 16-bit decryption result represents one number in the seed of each word in the plaintext. The seed is then processed using a distribution-transforming decoder to obtain the word representation of the seed. Once all the words have been obtained, all words are concatenated into one decrypted sentence. The equation used in the distribution-transforming decoder (DTD) from Honey Encryption ensures that the seed obtained from decryption always represents a word in the seed table (see Equation 18). Using an incorrect key for decryption can result in a seed (Seed') that is very different from the original seed. Therefore, the DTD process is important.

$$DTD = Seed' \ mod \ (max\_id\_on\_seed\_table + 1) \tag{18}$$

2.3. Evaluation Scenario

This experiment is evaluated from two perspectives: human and machine. In the human evaluation, there are two categorizations: human experts and laypersons who meet specific criteria. For machine evaluation, the Fine-Grained Evaluation of Dialogue (FED) method is used, which is an extension of DialoGPT [15].

Human Evaluation

Human evaluation uses four assessment dimensions: grammar, continuity, clarity, and ambiguity (see Table 2). There are two types of evaluation: layperson evaluations and expert evaluations. The results of the expert evaluation are used as a threshold for evaluating the contextual naturalness of the decoy messages. The laypersons evaluation result in this evaluation is the one that

represent the human evaluation. The smallest contextual naturalness score as the result of the expert's evaluation is the minimum score of the decoy message that is considered natural in the context. Equation 19 shows how the contextual naturalness score (*cns*) is calculated.

Table 2. Human Evaluation Dimensions

Dimension	Definition
Grammar	A grammar structure assessment is needed to determine whether the decrypted sentences follow the rules of grammar structure.
Continuity	Ensures that the decrypted result has context relation between chats during the conversation.
Clarity	Ensure clarity of understanding each chat in the conversation.
Ambiguity	This parameter is used to assess the number of ambiguous written sentences. The lower the ambiguity in the conversation, the higher the contextual naturalness of the conversation.

Based on the expert evaluation analysis, continuity and clarity have higher weight than grammar and ambiguity. In this case, continuity and clarity have a weight of 2, while grammar and ambiguity have a weight of 1, as shown in Equation 19.

$$cns = grammar\_score + (2 \times continuity\_score) + (2 \times clarity\_score) + ambiguity\_score \quad (19)$$

Machine Evaluation (FED)

FED uses DialoGPT to evaluate the dialogue. This method adds follow-up utterances to the last chat based on sentences categorized within existing dimensions. Table 3 lists some of the dimensions calculated by this method. After obtaining the evaluation score, dimensional mapping is carried out, referring to the assessment dimensions used in human evaluation. This mapping is based on the definition of each dimension in the machine evaluation. Once the mapping is complete, the contextual naturalness score of the decoy message is obtained using the same equation as in human evaluations (see Equation 19).

Table 3. FED and Human Evaluation Dimension mapping

Dimension	Definition	Human Dimension
Generic or Specific	In this domain, a generic or specific response is one that addresses the topic being discussed [16]	Continuity
Relevant	As the definition suggests, "relevant" means connected to the topic at hand [17]. Similarly, the relevant dimension is a category that assesses the level of connection between chats.	Continuity
Correct	In this dimension, FED evaluates the correlation level between the chat responses and the topic being discussed. Additionally, this dimension includes the level of misunderstanding in the chat.	Ambiguity
Semantically Appropriate	This dimension is also called "response-relateness." Evaluations in this dimension assess the appropriateness of responses between chats in the assessed dialogue [18]	Continuity
Understandable	Understandable means easy to understand [19]. In the context of dialogue, this dimension evaluates how easily the received message can be understood [15].	Clarity
Fluently Written	According to the definition, fluency means being able to easily use a language [20]. In this evaluation, the "fluency" dimension is used to assess the level of language proficiency in the chat responses [15].	Grammar
Coherent	In this dimension, coherence is evaluated based on how well the chat response continues the conversation and how clearly it describes things [21].	Clarity
Consistent	The purpose of this evaluation is to assess the consistency of the information conveyed in the conversation. This consistency can also be interpreted as harmony within the context of the evaluated chat [15].	Continuity
Diversity	The diversity dimension is a score reflecting the degree of difference in responses within the chat [22].	Ambiguity
Recipient Understanding	Similar to the understandable dimension, this dimension shows how the chat is perceived. The difference is that this dimension focuses on the recipient's point of view. This dimension assesses whether the recipient's response to the previous chat shows understanding of the previous message.	Clarity
Informative	Informative means that there is a lot of useful information [23]. In the FED application, this dimension represents an evaluation of chat content as being relevant to the topic of conversation.	Clarity

p-ISSN: 2528-1682
e-ISSN: 2527-9165

2. RESULT AND DISCUSSION

This section discusses the experimental results, contextual naturalness analysis of the bait message, and security analysis based on the probability of a successful key guessing attack. The contextual naturalness of the decoy messages is obtained by implementing KeyBERT and seed table generation using Electra MLM. This contextual naturalness makes it difficult for the attacker to guess both the correct plaintext and the correct key.

3.1. Contextual Naturalness Score

The results of this study were obtained by calculating a contextual naturalness score of the decoy message from both human and machine perspectives. The contextual naturalness can be used to distinguish the used key in the case of key guessing using brute force attack. The evaluated decoy message is categorized as natural based on the expert judgment. Natural decoy messages, as determined by experts, were analyzed by human and machine to determine the threshold score of contextual naturalness from both perspectives.

The analysis, based on both human and machine evaluations, revealed a minimum contextual naturalness score (the threshold) of **68.5972222222222%** for human evaluation and **79.757363484928%** for machine evaluation. The results of this study show that, from both perspectives, a total of 22 out of 32 evaluated decoy messages are categorized as contextual natural. Table 4 shows the contextual naturalness result of the evaluated decoy messages. The bold number represents the contextual naturalness score of the decoy message, which exceeds the minimum contextual naturalness score set by the expert. From the decoy messages resulting from the experiments, thirteen decoy messages were categorized as natural by human and machine evaluation.

In the machine evaluation, nine decoy messages were considered natural, but not by human evaluation. Most of the evaluated decoy messages have low scores in grammar. Based on the machine evaluation, the grammar of all decoy messages ranges from 55 to 57 percent. Nine decoy messages were considered natural by human evaluation, but also had a low score in grammar based on the machine evaluation. This low score refers to the limits of the grammar rules in the MLM dataset. The grammar score, based on human evaluation, falls within the range of 57% to 90%. Thus, the range of human evaluation is considered to be larger than that of machine evaluation. This condition occurred due to human subjectivity and differing perceptions in the implementation of grammar types in the chat.

The continuity evaluation result shows a better outcome than the grammar evaluation, as the average value of this dimension is greater than that of grammar. This score is higher than the grammar score due to keyword mapping, which ensures the decoy messages have a continuous context. Keyword mapping also affects the clarity dimension. Since keywords can represent the context of a sentence, they can increase the score of clarity. In the case of clarity evaluation, only two decoy messages have a score below the threshold value in both human and machine evaluation. The clarity evaluation results show that the clarity dimension has a significant contribution for generating a decoy message considered natural in the context. Based on the evaluation score, the context continuity and the clarity scores are high (above the threshold). This indicates that there is a context relation between chat messages, which is the main contribution of the proposed method.

Table 4. Experiment Result Comparison

Decoy Message	Human	FED	Decoy Message	Human	FED	Decoy Message	Human	FED
1	**83,44**	**79,92**	12	65,11	**79,79**	23	**69,77**	**79,75**
2	**72,72**	**79,77**	13	64,35	**79,79**	24	**72,88**	79,74
3	**74,58**	**79,80**	14	**78,33**	79,71	25	**73,74**	**79,82**
4	68,58	**79,91**	15	**69,87**	**79,88**	26	**69,66**	**79,85**
5	**89,07**	79,73	16	**74,53**	79,71	27	**68,59**	**79,77**
6	67,97	**79,82**	17	66,14	**79,85**	28	**72,69**	79,74
7	63,33	**79,92**	18	**69,61**	**79,76**	29	**83,72**	**79,77**

Decoy Message	Human	FED	Decoy Message	Human	FED	Decoy Message	Human	FED
8	**76,97**	**79,78**	19	**82,26**	79,72	30	65,80	**79,83**
9	**68,92**	**79,77**	20	**81,61**	79,74	31	**76,41**	79,71
10	66,63	**79,79**	21	**88,08**	79,68	32	64,58	79,74
11	64,60	**79,81**	22	**69,66**	**79,82**			

3.2. Security Analysis

Security analysis is conducted by calculating the probability of a successful key-guessing attack on the proposed method (P) as shown in Equation 20. This equation is calculated based on contextual naturalness ($P_{unnatural}$), keywords similarity ($P_{keyword\_leak}$), sentence similarity ($P_{sentence\_leak}$), and the encryption key itself ($P_{encryption\_key\_leak}$). In Equation 21, $P_{unnatural}$ represents the probability that the decoy message lacks naturalness in terms of context. This probability is calculated based on the number of chats that do not have contextual correlation with the previous chat. The greater the number of unnatural chats in a chat room, the greater the probability that a chat is a decoy. $P_{keyword\_leak}$ is the probability that the decoy message will have the same keyword as the original chat. This probability is calculated based on the total number of alternative keywords available (*total_keyword*) (see Equation 22). $P_{sentence\_leak}$ is the probability that the chat will contain the same sentences. This probability is calculated based on the total number of words in all the seed tables in the chat room (*twis_1, twis_2, twis_n*) as shown in Equation 23. Equation 24 represents the probability of finding the encryption key ($P_{encryption\_key\_leak}$). This equation is used because if the decryption produces the same chat as the plaintext, it does not mean that the used key is the same as the one used in the encryption process. Meanwhile, the probability of a key guessing attack in the previous method is $P_{previous} = P_{unnatural} \cdot P_{sentence\_leak} \cdot P_{encryption\_key\_leak}$. Thus, the probability of a successful guessing attack on the previous method is greater than that of the proposed method.

$$P = P_{unnatural} \cdot P_{keyword\_leak} \cdot P_{sentence\_leak} \cdot P_{encryption\_key\_leak} \tag{20}$$

$$P_{unnatural} = \frac{1}{total\_number\_of\_connected\_chat\_pairs} \tag{21}$$

$$P_{keyword\_leak} = \frac{1}{total\_keyword} \tag{22}$$

$$P_{sentence\_leak} = \frac{1}{twis\_1} \cdot \frac{1}{twis\_2} \cdots \frac{1}{twis\_n} \tag{23}$$

$$P_{encryption\_key\_leak} = \frac{1}{2^{128}} \tag{24}$$

3. CONCLUSION

This study addresses the limitation of Honey Encryption (HE) in instant messaging contexts, where the absence of correlation among decoy messages makes brute-force decryption results easy to analyze. Such inconsistencies enable adversaries to identify correct decryptions and infer the corresponding encryption keys.

To overcome this issue, a context-aware decryption enhancement is proposed, designed to obscure plaintext generated during brute-force attacks. By integrating keyword extraction and seed table generation based on a Masked Language Model (MLM), the proposed method produces decoy messages that maintain contextual coherence across conversations. This enhancement reduces the likelihood of adversaries discerning the original plaintext and encryption key from decryption attempts.

Experimental evaluation demonstrates that the proposed approach generates more contextually natural decoy messages than conventional HE methods, thereby strengthening encryption resilience against brute-force key guessing. Future work will focus on improving grammatical quality through fine-tuned MLMs with Grammatical Error Correction (GEC), as well as extending the method to multilingual environments and optimizing its performance for real-time communication systems. GEC can be used to evaluate the word list in the seed table obtained from MLM. By using this method, the system can detect grammatical errors in the word combinations it obtains and label those words with low naturalness values.

REFERENCES

[1] A. Gharbi and A. Nori, "Honey Encryption Security Techniques: A Review Paper," *AL-Rafidain J. Comput. Sci. Math.*, vol. 16, no. 1, pp. 1–14, 2022.
[2] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Phys. Nonlinear Phenom.*, vol. 404, p. 132306, 2020.
[3] H. D. Abubakar, M. Umar, and M. A. Bakale, "Sentiment classification: Review of text vectorization methods: Bag of words, Tf-Idf, Word2vec and Doc2vec," *SLU J. Sci. Technol.*, vol. 4, no. 1, pp. 27–33, 2022.
[4] Z. Cao, H. Yamada, S. Teufel, and T. Tokunaga, "Misalignment of semantic relation knowledge between WordNet and human intuition," in *Proceedings of the 13th Global Wordnet Conference*, 2025, pp. 25–36.
[5] A. E. Omolara and A. Jantan, "Modified honey encryption scheme for encoding natural language message," *Int. J. Electr. Comput. Eng. IJECE*, vol. 9, no. 3, pp. 1871–1878, 2019, doi: 10.11591/ijece.v9i3.pp1871-1878.
[6] E. O. Abiodun, A. Jantan, O. I. Abiodun, and H. Arshad, "Reinforcing the security of instant messaging systems using an enhanced honey encryption scheme: the case of WhatsApp," *Wirel. Pers. Commun.*, vol. 112, pp. 2533–2556, 2020.
[7] P. Manikandaprabhu and M. Samreetha, "A review of encryption and decryption of text using the AES algorithm," *Int. J. Sci. Res. Eng. Trends*, vol. 10, no. 2, 2024.
[8] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds., Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. doi: 10.18653/v1/D19-1410.
[9] P. Sharma and Y. Li, "Self-supervised contextual keyword and keyphrase retrieval with self-labelling," 2019.
[10] H. Face, "Masked Language Modeling." 2025. [Online]. Available: https://huggingface.co/docs/transformers/en/tasks/masked_language_modeling
[11] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators," in *ICLR*, 2020. [Online]. Available: https://openreview.net/pdf?id=r1xMH1BtvB
[12] A. K. Agarwal, L. Rani, R. G. Tiwari, T. Sharma, and P. K. Sarangi, "Honey encryption: Fortification beyond the brute-force impediment," in *Advances in Mechanical Engineering: Select Proceedings of CAMSE 2020*, Springer, 2021, pp. 673–681.
[13] N. F. Hassan, "Honey Encryption Techniques: Strengths, Limitations, and Challenges," in مجلة المنصور, vol. 42, no. 1, pp. 12–28, 2025.
[14] K. Sarmila and S. Manisekaran, "Honey encryption and AES based data protection against brute force attack," in *2022 Sixth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, IEEE, 2022, pp. 187–190.
[15] S. Mehri and M. Eskenazi, "Unsupervised Evaluation of Interactive Dialog with DialoGPT," in *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, O. Pietquin, S. Muresan, V. Chen, C. Kennington, D. Vandyke, N. Dethlefs, K. Inoue, E. Ekstedt, and S. Ultes, Eds., 1st virtual meeting: Association for Computational Linguistics, July 2020, pp. 225–235. doi: 10.18653/v1/2020.sigdial-1.28.
[16] G. Tyen, M. Brenchley, A. Caines, and P. Buttery, "Towards an open-domain chatbot for language practice," in *Proceedings of the 17th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2022)*, E. Kochmar, J. Burstein, A. Horbach, R. Laarmann-Quante, N. Madnani, A. Tack, V. Yaneva, Z. Yuan, and T. Zesch, Eds., Seattle, Washington: Association for Computational Linguistics, July 2022, pp. 234–249. doi: 10.18653/v1/2022.bea-1.28.
[17] Cambridge University Press, "Relevant." 2025. [Online]. Available: https://dictionary.cambridge.org/dictionary/english/relevant
[18] H. Zhang, H. Song, S. Li, M. Zhou, and D. Song, "A survey of controllable text generation using transformer-based pre-trained language models," *ACM Comput. Surv.*, vol. 56, no. 3, pp. 1–37, 2023.
[19] Cambridge University Press, "Understandable." 2025. [Online]. Available: https://dictionary.cambridge.org/dictionary/english/understandable
[20] Cambridge University Press, "Fluent." 2025. [Online]. Available: https://dictionary.cambridge.org/dictionary/english/fluent
[21] S. Zarrieß, H. Voigt, and S. Schüz, "Decoding methods in neural language generation: A survey," *Information*, vol. 12, no. 9, p. 355, 2021.
[22] W. Zhou, Q. Li, and C. Li, "Learning from Perturbations: Diverse and Informative Dialogue Generation with Inverse Adversarial Training," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, C. Zong, F. Xia, W. Li, and R. Navigli, Eds., Online: Association for Computational Linguistics, Aug. 2021, pp. 694–703. doi: 10.18653/v1/2021.acl-long.57.
[23] Cambridge University Press, "Informative." 2025. [Online]. Available: https://dictionary.cambridge.org/dictionary/english/informative