Visual Honey Encryption: Application to Steganography

Ji Won Yoon*
Center for Information Security Technologies (CIST)
Korea University
Republic of Korea
jiwon_yoon@korea.ac.kr

Hyoungshick Kim
College of Information and Communication Engineering
SungKyunKwan University
Republic of Korea
hyoung@skku.edu

Hyun-Ju Jo
Center for Information Security Technologies (CIST)
Korea University
Republic of Korea
heyonjulove@naver.com

Hyelim Lee
Center for Information Security Technologies (CIST)
Korea University
Republic of Korea
dream8933@naver.com

Kwangsu Lee
Center for Information Security Technologies (CIST)
Korea University
Republic of Korea
guspin.lee@gmail.com

ABSTRACT

Honey encryption (HE) is a new technique to overcome the weakness of conventional *password-based encryption* (PBE). However, conventional honey encryption still has the limitation that it works only for binary bit streams or integer sequences because it uses a fixed distribution-transforming encoder (DTE). In this paper, we propose a variant of honey encryption called *visual honey encryption* which employs an adaptive DTE in a Bayesian framework so that the proposed approach can be applied to more complex domains including images and videos. We applied this method to create a new steganography scheme which significantly improves the security level of traditional steganography.

Categories and Subject Descriptors

E.3 [**Data**]: Data Encryption

Keywords

Honey encryption (HE), Password-based encryption (PBE), Steganography

1. INTRODUCTION

Password-based encryption (PBE) schemes have many practical applications, particularly when the encryption and decryption keys should be memorized by the user. For example, PBE is used to protect Android phone owners' sensitive data; the disk or volume encryption key is derived from a user's screen-unlock password and a salt value [13]. Unlike biometrics and cryptographic keys, passwords are easy to implement and do not require additional hardware support. However, passwords too have their own inherent limitations – namely, memorability and security. A password that is difficult to guess is also likely to be hard to remember.

Thus, as one would imagine, many users tend to choose passwords that are easy to remember without really paying close attention to the security implications. Trivial passwords such as '`password`' and '`abcd1234`' are often used by casual users. As a result, the real password space is much smaller than the theoretical one [4, 11], making brute-force and dictionary attacks possible and effective. For example, a study of 544, 960 passwords collected from real users showed that the average entropy of user passwords was approximately 40.5 bits, smaller than the 128 or 192 bit standard used by many systems [6]. That is, when a PBE scheme is used, attackers may search amongst only a small subset of the theoretically possible passwords, the most commonly used ones, and crack encrypted information by guessing the password used to derive the encryption key [7].

Honey encryption (HE) was introduced to overcome limitations of PBE schemes [9, 8]. *Honey encryption* generates a ciphertext that can be decrypted with not only the correct password but also wrong passwords. In this scheme, wrong passwords can yield fake but valid-looking plaintext to confuse an attacker who is trying to decrypt the ciphertext through a brute force attack. Thus, the attacker is unable to identify the original plaintext even after trying all possible password combinations. Juels and Ristenpart [9] presented a general framework to construct a honey encryption scheme using a new (randomized) message encoding using *distribution-transforming encoders* (DTE) for binary bit stream and integers only. However, the scheme still has practical limitations because it uses a fixed simple DTE for encoding and decoding. That is, Juels and Ristenpart [9]'s honey encryption scheme cannot be flexibly used for numerous application domains including natural language-based texts, sounds, and images since such data has its their own synthetic or semantic structure. For example, a lot of multimedia data transferred in the Internet can be interpreted as a spatial grid structured field. Because of this image structure neighboring pixels have more similar colour than

*Yoon insisted his name be first and he is a corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
IH&MMSec'15, June 17–19, 2015, Portland, Oregon, USA.
Copyright © 2015 ACM 978-1-4503-3587-4/15/06 ...$15.00.
DOI: http://dx.doi.org/10.1145/2756601.2756606.

distant ones. Since such complicated data cannot be directly encrypted with original honey encryption, a new honey encryption system is needed for such structured data. We simply name such data *multi-dimensional data* in that images are well-known as being a two-dimensional data type and films can be regarded as a three-dimensional one.

In this paper, we introduce a new variation of honey encryption which can be applied to such complicated multidimensional data using a two-dimensional Markovian process in a Bayesian scheme. The Markovian process for 2D images is a well-known mathematical model for designing and interpreting the synthetic structure of images. In this paper, to distinguish it from original honey encryption, we name this algorithm *visual honey encryption* since we mainly consider visual 2D images. For a specific application domain using images, we made a connection between multi-dimensional honey encryption and steganography since steganography is one of the best-known applications which deal with images in security area. From this perspective, we introduce a new type of steganography based on *visual honey encryption*. In this paper, for simplicity, we also name this new steganography *honey steganography*.

Eventually, there are three contributions in this paper. First, we introduce an adaptive scheme of DTE for honey encryption. Thus, we do not need to fix or predetermine the DTE as conventional honey encryption does. The second contribution is that we introduce a new variation of honey encryption which removes the practical limitation and weakness of the original using adaptive DTE and a Bayesian framework design. Thereby, we can apply honey encryption to more complicated data including natural language-based texts and multimedia data. We name this new flexible honey encryption *visual honey encryption*. The final contribution is that we introduce a new steganography based on *visual honey encryption* which improves on the security of conventional steganography in terms of time complexity.

2. BACKGROUND

2.1 Honey Encryption

Honey encryption (HE), introduced by Juels and Ristepart [9], is a symmetric encryption scheme such as password-based encryption (PBE). In HE, the encryption algorithm encrypts a message into a randomized ciphertext by using a key, and the decryption algorithm decrypts the ciphertext into the original message by using the key in a similar manner to normal encryption schemes. The notable difference between honey encryption and conventional symmetric encryption is the output of the decryption algorithm when an invalid key is used; whereas conventional encryption gives an error symbol, HE gives a plausible message.

HE provides two security properties. If the keys used with HE are sufficiently unpredictable, HE provides semantic security, in which computationally bounded adversary cannot recover meaningful information from ciphertexts. Additionally, HE provides message recovery security, so that a computationally unbounded adversary who can decrypt the ciphertext by trying all possible keys is still unable to distinguish whether the recovered message is valid or not. In this way, HE can provide security against brute-force attacks. A general methodology for building HE schemes is to apply a distribution-transformation encoder (DTE) to a message and then encrypt the results of the encoding using a key such as a password. A DTE is a message encoding scheme and the DTE decoding algorithm can sample a message in the original distribution even when a random input value is given. Juels and Ristepart showed that credit card numbers and RSA private keys can be securely protected with HE.

2.2 Steganography

Cover modification is a well-known approach in steganography and a cover image embeds and conveys a desired secret message. Given the cover image, Alice modifies the cover image by embedding the hidden secret messages. The communication between Alice and Bob can be performed with the set of all possible cover images and the sets of the keys and messages. Let $\mathcal{C}$, $\mathcal{K}$, $\mathcal{M}$, and $\mathcal{S}$ denote a set of covers, a set of all stego keys, a set of all messages, and a set of stego images respectively. In general, since a steganographic scheme is a pair of embedding and extraction functions EMB and EXT, we have $EMB : \mathcal{C} \times \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{S}$, and $EXT : \mathcal{S} \times \mathcal{K} \rightarrow \mathcal{M}$ such that $EXT(EMB(\mathbf{c}, \mathbf{k}, \mathbf{m}), \mathbf{k}) = \mathbf{m}$ for all $\mathbf{c} \in \mathcal{C}$, $\mathbf{m} \in \mathcal{M}$, and $\mathbf{k} \in \mathcal{K}$. Here, let $\mathbf{s} = EMB(\mathbf{c}, \mathbf{k}, \mathbf{m})$ be the stego image for $\mathbf{s} \in \mathcal{S}$. In steganography and steganalysis, much research has been focused on creating perfectly secure steganography in which the characteristics of the distribution of the stego images p_s is that of distribution of cover images p_c. Kullback Leibler (KL) distance is a well-known measure for perfectly secure steganography and is defined by $D_{KL}(p_c||p_s) = \sum_{\mathbf{c} \in \mathcal{C}} p_c(\mathbf{c}) \log \frac{p_c(\mathbf{c})}{p_s(\mathbf{c})}$. Given the KL divergence, we can say that Alice's steganosystem is perfectly secure when $D_{KL}(p_c||p_s) = 0$. In this case, the malicious person Eve cannot distinguish between cover images and stego images. However, it is often impractical to build such perfectly secure steganography and many practical stegano-systems are not perfectly secure. Nevertheless they are ϵ-secure with the constraint $D_{KL}(p_c||p_s) < \epsilon$. In this paper, we show three additional measures to KL divergence: root mean square error (RMSE), peak signal to noise ratio (PSNR), and structural similarity (SSIM) [16].

2.3 Multi-dimensional Data

We define some terms related to multi-dimensional data for better understanding of the difference between conventional HE and our proposed scheme. Thus, in this paper, *One dimensional data* means both binary bit streams like '1011001' and integer sequences like '01028553945'. *Multi-dimensional data* means more complicated data including images, natural language based texts and films.

To date, HE handles only one dimensional data and generates one dimensional honeywords (fake data). However, as computer performance increases, multi-dimensional data becomes more popular. Multi-dimensional data, which include images and films, is the most common form of multimedia data, but cannot be encrypted with conventional honey encryption as it can only handle on-dimensional data. Thus, multi-dimensional honey encryption should be developed for honey encryption to become applicable to most multimedia data. In order to distinguish between conventional one dimensional honey encryption and our proposed approach, in this paper, we name the proposed approach *Visual honey encryption* (VHE).

In VHE, a honey multimedia data set generated via the encoding operation under encryption with incorrect passwords cannot be distinguished from the original multimedia

datum, which can be decrypted and decoded with the correct key alone.

3. THREAT MODEL

In this section, we discuss the threat model and assumptions.

The adversary can access the encrypted data and has full knowledge of our encryption scheme, but lacks the key used for encrypting data. We assume a computationally bounded adversary running in a polynomial time, incapable of breaking the encryption algorithm without knowing the key used for encryption; however, the adversary is capable of breaking a steganographic scheme in a limited time $T(n)$ where n is the size of the image being used. This assumption is reasonable since most steganographic techniques have been broken by effective steganalysis given enough resources [1] while breaking advanced encryption algorithms (e.g., AES [5]) is computationally infeasible for even the most powerful supercomputers.

We also assume that the encryption key used for our scheme is derived from a user-chosen password. This is a common method in practice when using encryption algorithms for user applications. In particular, Password-Based Key Derivation Function 2 (PBKDF2) [10] is popularly used as a key derivation function that is part of RSA Laboratories' Public-Key Cryptography Standards (PKCS) series.

However, in practice, many users choose passwords that are easy to remember without paying close attention to the security implications. Therefore, the actual password space used is much smaller than the theoretical one [4, 11], and this dramatically increases the likelihood of attacker compromising a password through guessing attacks. That is, we assume the adversary can iteratively guess the user-chosen password and try to derive the encryption key to decrypt the encrypted data. We use Ψ to represent the actual password space which is much smaller than the theoretical password space. We note that $|\Psi|$ represents the size of the actual password space.

Given the above threat model and assumption, our goal is to protect the user's secret message in the encrypted image so that the adversary only knows the presence of the encrypted data and its characteristics (e.g., creation time, size, etc.) but not the secret message itself.

4. PROPOSED APPROACH

In this section, we first demonstrate the main concept and structure of VHE. Then, we present a new procedure to build VHE's DTE to accommodate multi-dimensional data. Afterwards, we show how to encode and decode the values using the VHE's DTE.

4.1 Concepts

VHE encodes and decodes senders' and receivers' data using a codebook that is extracted from the statistical properties of the multi-dimensional data. In this process, only authorized users with correct passwords or keys are able to obtain correct data (images/videos). In contrast, non-authorized users without correct passwords cannot obtain the information about the original data. Instead, they will obtain a wrong but valid-looking multi-dimensional datum that is generated by the rules of our statistical codebook and DTE.

Without loss of generality, in this paper, we particularly focus on applying VHE to images because they are representative of multi-dimensional data type. Note that our VHE does not encrypt and encode the header information of the multimedia but the internal pixel values only. From this point of view, our proposed VHE can be regarded as a file encryption system rather than channel encryption although VHE is also considered for communication channels. In addition, unlike text data, pixels have a high degree of similarity. That is, pixel values of real images are not random but highly connected with their neighbors. This neighboring property provides many useful mathematical properties of the image of practical use. For example, a Markov random field (MRF) is used for modeling such a neighboring structure in order to reduce the time complexity of image computation. In this paper, given this mathematical property, we make an adaptive DTE from the images using the Markovian rule.

4.2 Structure of VHE

VHE consists of a sequential processes: (1) selection of data, (2) construction of a codebook using statistical formula, (3) encoding and decoding using the codebook, (4) encryption with a key/password K_1, and (5) transmission of the encrypted message. Each process is described in details:

1. **Selection of data**: To begin with, we select two types of images for a plain image, $\mathbf{p}$ and d_c public images, $\mathbf{Y}$s. The plain image has the same format and size as the fake images. The plain image is a hidden information while the fake images are in public. The public images are shared between Alice and Bob in order to use them to construct a DTE. This is known even to Eve who is a malicious subject. Since the encoder and decoder should use an identical DTE, the fake images should be shared between Alice and Bob before communication.

2. **Construction of a codebook using statistical formula**: Let $\mathbf{x}$, $\mathbf{Y}$, and θ denote the encoding space, a set of public images, and other public parameters of the model. In this stage, using the selected image $\mathbf{Y}$ and other known parameters, VHE constructs a full joint target posterior distribution $p(\mathbf{x}|\mathbf{Y}, \theta)$ for a statistical codebook and the distribution is used as a DTE is in conventional honey encryption. VHE systematically uses a conditional posterior distribution, $p(x_i|\mathbf{x}_{ne(i)}, \mathbf{Y}, \theta)$, rather than the full joint target distribution since every pixel is sequentially coded in a pixel-by-pixel order where $ne(i)$ denotes a set of indexes of the ith pixel's neighbors. As soon as the conditional posterior of the i-th pixel is constructed, VHE builds the corresponding cumulative mass function (CMF) of the conditional target posterior. Let $p_i \in \{0,1\}^{d_p}$ and $c_i \in \{0,1\}^{d_c}$ be the i-th plain image and the i-th encoded data respectively. For simplicity, d_c becomes either 8 for gray-scale images or 24 for true color images.

3. **Encoding and decoding using the codebook**: In the encoding and decoding stages, VHE uses the statistical codebook, CMF$(\cdot)$, to encode p_i into c_i or to decode c_i into p_i. This CMF works as a statistical

codebook by

$$\begin{aligned} c_i &= \text{CMF}(p_i) \text{ , for encoding and } c_i \in \{0,1\}^{d_c} \\ p_i &= \text{CMF}^{-1}(c_i) \text{ , for decoding } p_i \in \{0,1\}^{d_p} \quad (1) \end{aligned}$$

where $d_p \leq d_c$. That is, input values with d_p binary digits are encoded into d_c binary digits via CMF$(\cdot)$ and CMF$^{-1}()$ is the inverse operation. Therefore, each pixel value in a plain image can be encoded or decoded in terms of the statistical properties of the shared public images.

4. **Encryption with an encryption key K_1 and decryption with a decryption key K_2**: Encryption is performed using conventional encryption algorithms like AES, RSA, and so on. If $K_2 = K_1$ in the symmetric crypto-system for an appropriate receiver, Bob, the encrypted cipher will be correctly decrypted and then decoded to the hidden plain image $\mathbf{p}$ via our DTE. Otherwise, the encrypted cipher will be decrypted to a random sequence which is different from the actual bit-streams because $K_1 \neq K_2$ and the malicious subject Eve obtains variations of fake images $\mathbf{h}$ which are decoded from the random sequence.

5. **Transmission of the encrypted data**

For instance, we have a hidden plain image $\mathbf{p}$ of Figure 1-(a) and a public image $\mathbf{Y}$ of Figure 1-(b). As shown in Figure 1-(c) and -(d), for a symmetric encryption system, the receiver can obtain a correct hidden image when K_1 and K_2 are identical. However, the receiver will obtain a number of fake images with some variation when $K_1 \neq K_2$. That is, there is only one $\mathbf{p}$ but there are $|\mathcal{K}_{enc/dec}|$ $\mathbf{h}$ where $|\mathcal{K}_{enc/dec}|$ is the cardinality of the key or password space for encryption or decryption for $K_1 \in \mathcal{K}_{enc}$ and $K_2 \in \mathcal{K}_{dec}$.

(a) $\mathbf{p}$ (b) $\mathbf{Y}$ (c) $\mathbf{p}$ if $K_1 = K_2$ (d) $\mathbf{h}$ if $K_1 \neq K_2$

Figure 1: Input and output of pseudo-VHE where K_1 and K_2 are the passwords for encryption and decryption in a symmetric crypto-system.

4.3 Connection to Steganography

Figure 1 shows that, when the cipher is decrypted and decoded with incorrect passwords, Eve obtains a false image which has similar properties to the desired real image. The power of visual honey encryption (VHE) is that Eve does not obtain meaningless random images at all. However, the VHE may not be useful in practice for two reasons: 1) Figures 1-(c) and -(d) are different, their histograms will be different so, a machine will be able to distinguish them; and 2) although there are various fake or deception images $\mathbf{h}$ as shown in Figure 1-(d), they are much closer to the public image $\mathbf{Y}$ than the original plain image $\mathbf{p}$ of Figure 1-(c). Therefore, our proposed visual honey encryption is not perfect honey encryption. However, we realized that there is a simple but practical solution to address this problem. The solution is that $\mathbf{p}$ is replaced by $\mathbf{Y}$. In this case, the difference between the receiver's outputs $\mathbf{p}$ and $\mathbf{h}$ is dramatically reduced such that $|\mathbf{p} - \mathbf{h}|$ of Figure 2 is smaller than $|\mathbf{p} - \mathbf{h}|$ of Figure 1.

(a) $\mathbf{p}$ (b) $\mathbf{Y}$ (c) $\mathbf{p}$ if $K_1 = K_2$ (d) $\mathbf{h}$ if $K_1 \neq K_2$

Figure 2: Input and output of practical VHE

Figure 2 shows how to make practical VHE. However, in this case, we cannot select a hidden image to be sent by ourselves as shown in Figure 1 since $\mathbf{p}$ should be a variation of public images $\mathbf{Y}$, i.e., $\mathbf{p} \approx \mathbf{Y}$. Interestingly, we found that this practical shortcoming of VHE can be removed by combining it with steganography. We can embed a secret message $\mathbf{m}$ into $\mathbf{p}$ which now becomes a cover image $\mathbf{c}$, i.e., $\mathbf{s}$=Emb($\mathbf{c}$, $\mathbf{m}$). In this case, $\mathbf{p} = \mathbf{s} \approx \mathbf{Y}$ and $\mathbf{m}$ =Ext($\mathbf{s}$) but $\mathbf{m}$ $\neq$Ext($\mathbf{h}$).

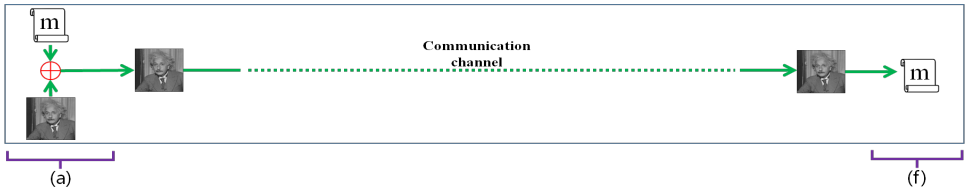
With this combination of steganography and practical VHE, a new powerful steganography algorithm is introduced as shown in Figure 3. The Figure presents three different models of steganography. Figure 3-(1) is the traditional steganography model in which the stego-image is directly transmitted to Bob without any encryption. The attacker performs only steganalysis to extract the hidden message with the stego-image. Figure 3-(2) is more complicated model than (1) since an encryption process is added during communication. Although in practice many systems follow this model as communication channels are increasingly encrypted, it contains no theoretical improvements in either steganography or encryption, and has therefore not been widely discussed in the literature. Figure 3-(3) depicts our proposed model which is a variation of Figure 3-(2) in which the traditional ASCII encoder and decoder are replaced by our proposed DTE based encoder and decoder.

4.4 Construction of Encoding/Decoding Distribution

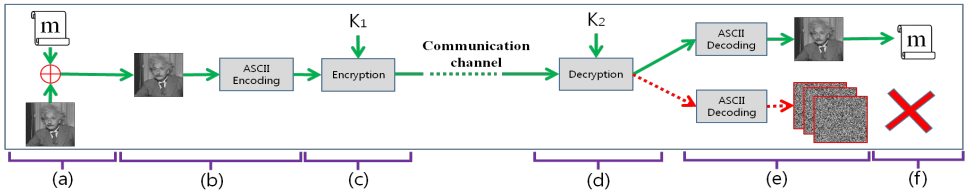
Before describing the main algorithm of the new steganography scheme based on VHE as depicted in Figure 3-(3), we define the symbols which are used in the algorithm.

In this table, $\mathbf{p}$ for a stego image and $\mathbf{Y}$ for a fake image are matrices of $L_1 \times L_2$ size, and $\mathbf{x}$ and $\mathbf{y}$ are their corresponding vectorized forms with $L_1 L_2 \times 1$ for $L = L_1 L_2$.

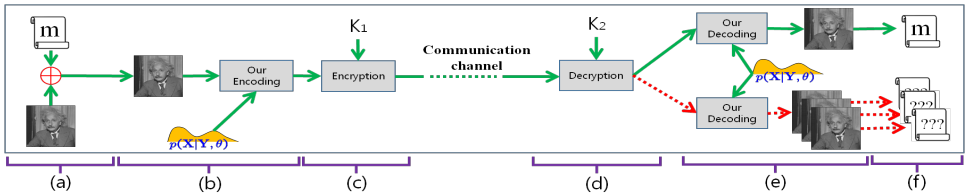
The underlying concept behind our proposed approach is the use of statistical coding schemes instead of the traditional ASCII coding schemes. Given a stego-image $\mathbf{s}$ and a fake image $\mathbf{Y}$, we can make encoding or decoding procedures by reconstructing the underlying probability density function. In this paper, denote $p(\mathbf{x}|\mathbf{Y}, \theta)$ by the density function, similar to the DTE of Honey Encryption. This distribution can be interpreted well in a Bayesian framework. For a simplified representation, we vectorized the matrices to build $\mathbf{x}$ and $\mathbf{y}$, transforming $\mathbf{x} = \mathcal{V}(\mathbf{s})$ and $\mathbf{y}^{(n)} = \mathcal{V}(\mathbf{Y}^{(n)})$, which are more familiar forms for conventional Bayesian statistics. In this vectorized form $\mathbf{x}$ denotes $\mathbf{x}_{1:L_1 L_2} = \{x_i\}_{i=1}^{L_1 L_2}$. By



(1) Steganography without encryption module



(2) Steganography with encryption module



(3) Proposed Honey steganography

Figure 3: Various steganography schemes: (1) Steganography without an encryption module, (2) Steganography with an encryption module, (3) Proposed Honey steganography. In this Figure, green solid arrows represent the flow of appropriate communication between Alice and Bob while red dotted arrows represent an inappropriate flow between Alice and Eve. There are six steps in the Figure: (a) embedding messages to cover image, (b) encoding, (c) encryption, (d) decryption, (e) decoding, and (f) extracting messages.

Table 1: Definition of the used symbols

Symbols	Definition
N	the number of public images or duplicates of a public image
d_p	the number of binary bits of plain texts for each operation
d_c	the number of binary bits of cipher texts for each operation
$\mathbf{m}$	a secret message to be sent
$\mathbf{c}$	a cover image in which the secret message will be embeded
$\mathbf{s}$	Stego-image with $\mathbf{c}$ and $\mathbf{m}$
$\mathbf{z}$	Encoded data from $\mathbf{s}$
$\mathbf{p}$	a hidden stego image to be sent, $\mathbf{p} = \mathbf{s}$
$\mathbf{Y}$	public image(s)
$\mathcal{V}(\cdot)$	a transformation function from matrix to vector
$\mathbf{x}$	a vectorized form that encodes $\mathbf{s}$
$\mathbf{y}$	a vectorized form that encodes $\mathbf{Y}$
$\mathbf{u}$	encrypted data

using the Bayesian chain rule, we now have

$$p(\mathbf{x}|\mathbf{y},\theta) = \prod_{i=1}^{L} p(x_i|\mathbf{x}_{1:i-1},\mathbf{y},\theta) \tag{2}$$

where $L = L_1 L_2$. In general, images have a special grid structure and this is often modeled with a two dimensional the Markovian structure because it can reduce the time and space complexity of the computation using Markovian blankets. Therefore, we have a target distribution and it is factorized as

$$p(\mathbf{x}|\mathbf{y},\theta) = \prod_{i=1}^{L} p(x_i|\mathbf{x}_{MB(i)},\mathbf{y},\theta) = \prod_{i=1}^{L} p(x_i|\mathbf{x}_{ne(i)},\mathbf{y},\theta). \tag{3}$$

Here, x_i is the i th pixel value. $MB(i)$ and $ne(i)$ denote Markov blanket and neighbors of the ith pixel for dependency. Note that our proposed approach performs the encoding and decoding procedures sequentially. Therefore, our immediate goal is to construct the conditional distribution of the ith pixel instead of the full joint distribution of a full image, $p(x_i|\mathbf{x}_{ne(i)},\mathbf{y})$. This conditional distribution means that the ith pixel is influenced by neighboring pixels and values of the fake image.

The conditional density of the ith pixel can be further reduced by assuming independence of pixels in the fake image such that

$$p(x_i|\mathbf{X}_{ne(i)},\mathbf{y},\theta) \quad = \quad p(x_i|\mathbf{x}_{ne(i)},y_i,\mathbf{y}_{\sim i},\theta) \tag{4}$$

where y_i is the ith pixel of the clean fake image corresponding to x_i and $\mathbf{y}_{\sim i}$ denotes the vectorized form of $\mathbf{y}$ except y_i. That is, $\mathbf{y} = y_i \cup \mathbf{y}_{\sim i}$ and $y_i \cap \mathbf{y}_{\sim i} = \{\}$. Now this

distribution can be rewritten by

$$p(x_i|\mathbf{X}_{ne(i)},\mathbf{y},\theta) = p(x_i|\mathbf{x}_{ne(i)},y_i,\mathbf{y}_{\sim i},\theta) \propto p(y_i|x_i,\theta)p(x_i|\mathbf{x}_{ne(i)},\theta). \quad (5)$$

In Equation (5), we now have two key factors: likelihood function $p(y_i|x_i,\theta)$ and the prior function $p(x_i|\mathbf{x}_{ne(i)},\theta)$. These are explained in a Bayesian framework. The first factor is the likelihood function $p(y_i|x_i,\theta)$ that is the probability of how x_i well fits the fake pixel y_i. The other factor is a prior term that is constructed with neighboring values. In the image application, this prior is often designed with a Markov random field [2, 3, 12].

Note that equations (2) and (5) are basically targeted to a single fake image $\mathbf{Y}$ and its vectorized form $\mathbf{y}$. In this single image case, the prior and likelihood can have exactly the same influence on building the posterior. However, we could have multiple fake images instead of a single one. In this case, the equations can be rewritten by $\mathbf{Y} = \mathbf{y}^{(1:N)} = \mathbf{y}^{(1)},\mathbf{y}^{(2)},\cdots,\mathbf{y}^{(N)}$:

$$p(\mathbf{x}|\mathbf{Y},\theta) = \prod_{i=1}^{L} p(x_i|\mathbf{x}_{ne(i)},\mathbf{Y},\theta) \propto \prod_{i=1}^{L}\left[\prod_{n=1}^{N} p(y_i^{(n)}|x_i,\theta)\right] p(x_i|\mathbf{x}_{ne(i)},\theta) \quad (6)$$

where $y_i^{(n)}$ represents the ith pixel value of the nth fake image. Returning to the conditional posterior of the ith pixel for the multiple fake images, we have $p(x_i|\mathbf{x}_{ne(i)},\mathbf{Y},\theta) = \left[\prod_{n=1}^{N} p(y_i^{(n)}|x_i,\theta)\right] p(x_i|\mathbf{x}_{ne(i)},\theta)$. In this paper, we define the likelihood and prior using well-known normal distribution by

$$p(y_i^{(n)}|x_i,\theta) = \mathcal{N}(y_i^{(n)};x_i,r^2) = \mathcal{N}(x_i;y_i^{(n)},r^2)$$
$$p(x_i|\mathbf{x}_{ne(i)},\theta) = \mathcal{N}(x_i;f(\mathbf{x}_{ne(i)}),\rho^2) \quad (7)$$

where $\sigma \in \theta$ and $\rho \in \theta$ are the standard deviations of each distribution and $\mathcal{N}(\cdot;a,b)$ is the normal distribution with a mean a and a variance b. $f(\cdot)$ is any linear/nonlinear function.

It is known that the product of the normal distributions becomes a normal distribution as shown in appendix A. Therefore, $p(x_i|\mathbf{x}_{ne(i)},\mathbf{Y},\theta)$ from equation (7) is unified in a collapsed normal distribution by

$$p(x_i|\mathbf{x}_{ne(i)},\mathbf{Y},\theta) = \mathcal{N}(x_i;\mu,\sigma^2) \quad (8)$$

where

$$\sigma = \sqrt{\left(\sum_{n=1}^{N}\frac{1}{r^2}+\frac{1}{\rho^2}\right)^{-1}}$$
$$\mu = \sigma^2\left(\frac{\sum_{n=1}^{N} y_i^{(n)}}{r^2}+\frac{f(\mathbf{x}_{ne(i)})}{\rho^2}\right).$$

In addition, one fake image can be used for multiple fake images by duplicating it to N copies. In this paper, we make N images using one fake image in this way. In the result section, we show that the degree of influence varies as N varies.

4.5 Encoding and decoding schemes

Each pixel has 8 bits for single channel or gray-scale images so $x_i \in \{0,1,\cdots,2^8-1\}$. The basic idea of honey encryption is to change the encoding and decoding rules from traditional ASCII to distribution based coding scheme. Each value can be encoded and decoded with different weights in the statistical coding scheme. This means that some values can have more weight than others. In order to provide these varying weights, the length of the encoded data should be increased. In other words, in order to encode values in a d_c bit system ($d_c = 8$ for image pixels), longer bits are required to encode the value, i.e. $d_p \le d_c$. Now let's return to our single channel image. In this case, we encode 2^{d_p} values into 2^{d_c} binary codes. This also means that d_c bit images are quantized to d_p bit pixels. In the results section, we set $d_p = 4$ and $d_c = 8$. The set of 2^{d_p} values can be defined manually or automatically by $\mathcal{Z} = \{z_0 \cup z_1 \cup ... \cup z_{2^{d_p}-1}\}$. In general, $\min \mathcal{Z} \le 0$ and $2^{d_c} > \max \mathcal{Z}$ for d_c bit images. We first discretize and quantize the conditional posterior distribution $p(x_i = z|\mathbf{x}_{ne(i)},\mathbf{Y},\theta)$ of equation (8) to represent the probability masses function (PMF) of the $2^d = 16$ discrete values for $z \in \mathcal{Q}$. Now the cumulative mass function of the ith pixel is defined by

$$p_{cmf}^{(i)}(z_k) = \sum_{k=0}^{2^k-1} \frac{p(x_i = z_k|\mathbf{x}_{ne(i)},\mathbf{Y},\theta)}{\sum_{j=0}^{2^d-1} p(x_i = z_j|\mathbf{x}_{ne(i)},\mathbf{Y},\theta)}. \quad (9)$$

Before encoding or decoding the values using the CMF of the equation (9), we need to modify the CMF to maintain the consistency of the encoders and decoders. As already mentioned, 2^{d_p} values(symbols) are encoded to corresponding d_c-digit values with different weights. Symbols with a higher PMF. will cover more values among 2^{d_c} values and some symbols with a relatively low PMF will cover less. These are the characteristics of DTE of honey encryption which cause its power. However, they cause inconsistencies in the encoding and decoding operations. If the PMF of some symbols are lower than $1/2^{d_c}$, then we cannot encode them. Therefore, we need to change the PMF and CMF to ensure that the PMF of any symbol is larger than $1/2^{d_c}$. In order to achieve this, we reduce the probability of symbols with dominant weights and the removed weights are added to other symbols with low weights.

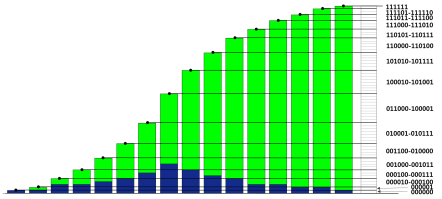


Figure 4: PMF (dark blue bars) and CMF (bright green bars) for encoding and decoding. For $d_p = 4$ and $d_c = 6$, 16 symbols with 4 bits digit are encoded into 6 bit digits.

Figure 4 displays three informative plots about an encoder from d_p bit digits to d_c bit digits and a decoder from d_c digits to d_p digits. The dark blue bars represent the probability mass function of $p(x_i|\mathbf{x}_{ne(i)},\mathbf{Y},\theta)$ of equation (8) and the green bars represent their corresponding CMF. Last, each bar of CMF indicates the d_c bit codes. That is, the 2^{d_p} values of $p(i)$ are encoded to 2^{d_c} binary digits of c_i via $c_i = CMF(p_i)$. For instance, we set $d_p = 4$ and $d_c = 6$ for simplicity in Figure 4.

4.6 Necessity of pre-processing

In the previous section, we described the process of the cumulative mass function (CMF) and the way to assign the bit array. Assigning the bit array to each case seems easy, but for exact encoding and decoding there is one problem to solve; every case must be matched with at least one bit array. In practice, there are various probabilities and some are really small values or extremely dominating. However even a small value has to be presented with at least one bit array. If a small value is not matched to one bit array then we cannot express it. Therefore, in this step, we need to assign a one bit array to even negligibly small probabilities. This step is necessary but it changes probabilities of cases and the CMF is also modified. The probabilities with smaller values than $1/2^{d_c}$ are reassigned to $1/2^{d_c}$ and dominating probabilities have their values reduced. Since small probability cases are assigned the probability $1/2^{d_c}$, an output image $\mathbf{h}$ that has been decrypted and decoded with an incorrect key has shot noise.Therefore, unfortunately, the attacker can distinguish between $\mathbf{p}$ and $\mathbf{h}$s easily. We tackle this problem, as shown in Figure 5, by adding a pre-processing step in which the sender's cover image $\mathbf{c}$ is replaced by one of $\mathbf{h}$s obtained by decoding a random image $\mathbf{r}$. Then, the receiver's $\mathbf{s}$ and $\mathbf{h}$ can be distinguished neither by human beings nor by machine, satisfying the property of honey encryption with ϵ-bound, i.e., $|\mathbf{s} - \mathbf{h}| - \epsilon < 0$ for an extremely small ϵ. Finally, we can make a pseudo-algorithm for our honey steganography, algorithm 1 for Alice and algorithm 2 for Bob and Eve as shown.

Algorithm 1 Sender's view of honey steganography

Require: Public: shared clean images $\mathbf{Y}$
Require: Private: a message $\mathbf{m}$ and an encryption key K_1.
1: Generate a random image $\mathbf{r} \in \mathcal{R}$.
2: **for** $i = 1$ to L **do**
3: Infer a conditional posterior $p(x_i|\mathbf{x}_{ne(i)}, \mathbf{Y}, \theta)$.
4: Obtain a CMF(i) by cumulating the posterior.
5: Build DTE(i) for encoder and decoder using CMF(i).
6: c_i =Decode(DTE(i), r_i) ▷ Decode $\mathbf{r}$ into $\mathbf{c} \in \mathcal{C}$.
7: **end for**
8: $\mathbf{s} = \mathbf{c} \oplus \mathbf{m}$ ▷ Make a stego-image $\mathbf{s} \in \mathcal{C}$ with $\mathbf{m}$ and $\mathbf{c}$.
9: $\mathbf{z}$ =Encode(DTE, $\mathbf{s}$) ▷ Encode the stego-image $\mathbf{s}$.
10: $\mathbf{u}$ =Encrypt($\mathbf{z}$, K_1) ▷ Encrypt the encoded sequence with a secret key K_1.

5. RESULTS

5.1 Description of Data

Top (Einstein), middle (Rome), and bottom (Lena) of Figure 6-(a) are the raw images used in this paper. We let $\mathbf{I}_{Einstein}$, $\mathbf{I}_{Rome}$, and $\mathbf{I}_{Lena}$ denote Einstein, Rome, and Lena images respectively. The size of every image is 128×128, i.e. $L_1 = 128$ and $L_2 = 128$ and they are all grayscale images because only the red channel of the true colors is used.

5.2 Parameters Used in Experiments

In the simulation, we simplify $f(\mathbf{x}_{ne(i)})$ to a running average filter for the Gaussian Markov random field (GMRF) by $f(\mathbf{x}_{ne(i)}) = \frac{1}{|ne(i)|}\sum_{j\in ne(i)} x_j$ to satisfy $x_i = \frac{1}{|ne(i)|}\sum_{j\in ne(i)} x_j + \epsilon_i$ where noise $\epsilon_i \sim \mathcal{N}(\cdot; 0, \rho^2)$ and $|\cdot|$ is the cardinality

Algorithm 2 Receiver's view of honey steganography

Require: Public: shared clean images $\mathbf{Y}$ and encrypted message $\mathbf{u}$
Require: Private: an decryption key K_2.
1: **for** $i = 1$ to L **do**
2: Infer a conditional posterior $p(x_i|\mathbf{x}_{ne(i)}, \mathbf{Y}, \theta)$.
3: Obtain a CMF by cumulating the posterior.
4: Build DTE for encoder and decoder using CMF.
5: **end for**
6: **if** K_2 is a correct decryption key **then**
7: $\mathbf{z}$ =Decrypt($\mathbf{u}$, K_2) ▷ Decryption
8: $\mathbf{s}$ =Decode(DTE, $\mathbf{z}$) ▷ Decoding
9: $\mathbf{m}$ =Extraction($\mathbf{s}$) ▷ Extraction from stego-images
10: **else**
11: $\tilde{\mathbf{z}}$ =Decrypt($\mathbf{u}$, K_2) ▷ Decryption
12: $\tilde{\mathbf{h}}$ =Decode(DTE, $\tilde{\mathbf{z}}$) ▷ Decoding
13: Apply steganalysis to extract $\mathbf{m}$ from $\tilde{\mathbf{h}}$

Extraction($\mathbf{h}$) ▷ Extraction from stego-images

14: **end if**

of a set. With this model, we fix $r \in \theta$ and $\rho \in \theta$ of equation (7) at $r = 1$ and $\rho = \sqrt{1/|ne(i)|}$ for simplicity. Then, the conditional posterior distribution of our interest is formed by $p(x_i|\mathbf{x}_{ne(i)}, \mathbf{Y}, \theta) = \mathcal{N}(x_i; \mu_i, \sigma_i^2)$ where $\sigma_i = (N+|ne(i)|)^{-1/2}$ and $\mu_i = (N+|ne(i)|)^{-1}\left(\sum_{n=1}^{N} y_i^{(n)} + \sum_{j\in ne(i)} x_j\right)$.

5.3 Evaluation Metrics

There is an important issue to be addressed to validate our proposed approach. Since this honey steganography also inherits the characteristics of honey encryption, for an ϵ-secure stegosystem, we need to check that the images obtained with incorrect keys or passwords are machine indistinguishable from one obtained with correct keys/passwords. There are various metrics to measure this based on similarities or differences between the images: Kullback Leibler distance (KLD), Peak Signal to Noise Ratio (PSNR), Root Mean square error (RMSE), and structural similarity (SSIM) [15, 14]. The details of the similarity measures are described in [16]. From now on we set $type \in \{KLD, PSNR, RMSE, SSIM\}$. Given these metrics, there are two different cases that should be evaluated:

- $\mathcal{D}_{type}(\mathbf{Y}, \mathbf{s})$ and $\{\mathcal{D}_{type}(\mathbf{Y}, \mathbf{h}^{(j)})\}_{j=1}^{R}$: this is the set of distance between a public image $\mathbf{Y}$ and a decoded stego-image with a correct decryption $\mathbf{s}$ and decoded honey images with R incorrect decryption, $\mathbf{h}^{(1:R)}$.
- $\{\mathcal{D}_{type}(\mathbf{s}, \mathbf{h}^{(j)})\}_{j=1}^{R}$ and $\{\mathcal{D}_{type}(\mathbf{h}^{(i)}, \mathbf{h}^{(j)})\}_{i,j=1, i\neq j}^{R}$: this is the set of the distance between decoded images.

Given this setting, we estimate the p-values of each metric to evaluate the distinguishability.

5.4 Steganography with Visual Honey Encryption (Honey Steganography)

Figure 6 demonstrates several input and output images obtained from honey steganography of three images. Figure 6-(a), (b), and (c) are handled by a sender, Alice. The other sub-figures can be obtained by two different types of receivers: an appropriate receiver Bob of Figure 6-(d) and a malicious receiver Eve of Figure 6-(e).

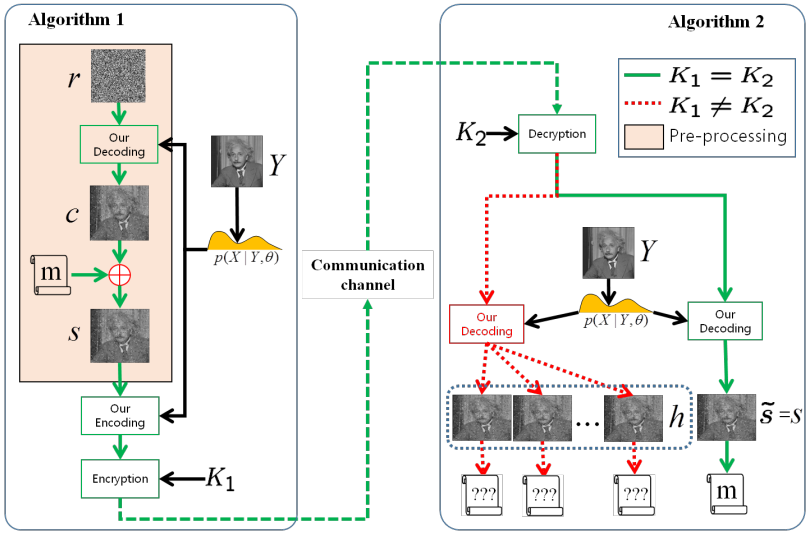


Figure 5: Proposed approach satisfies the indistinguishableness of images decrypted and decoded with a correct key and with incorrect passwords when a pre-processing procedure is adopted. Here, Y is a public image.

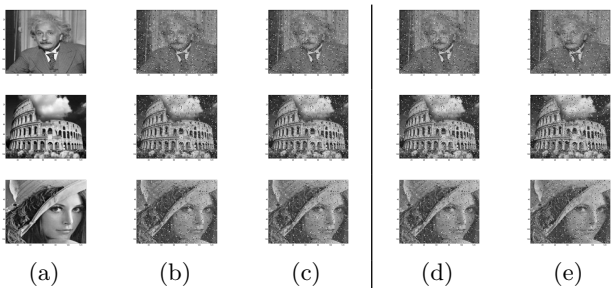


Figure 6: Procedure of VHE for three examples: (a)→(b)→(c)→(d) with a correct key and (a)→(b)→(c)→(e) with an incorrect key. Each columns are (a) public image Y, (b) noisy cover image c, (c) noisy stego image s, (d) correctly decoded image $\tilde{s}$, and (e) incorrectly decoded image h

The three images of Figure 6-(a) are the public clean images $\mathbf{Y}$ used to construct DTE with inferred conditional target posterior $p(x_i|\mathbf{x}_{ne(i)}, \mathbf{Y}, \theta)$ for $i \in \{1, 2, \cdots, L\}$. As we can see the images, clean images are used in this paper although, in practice, $\mathbf{Y}$ could be noisy or random images. In order to build DTE, $\mathbf{Y}$ is shared between Alice and Bob and it does not have to be shared privately. Therefore, $\mathbf{Y}$ is public information and even the malicious user Eve can access it. Figure 6-(b) demonstrates the three cover images obtained by decoding random images using our DTE. That is, we have $\mathbf{c}$ =Decode(DTE, $\mathbf{r}$) where $\mathbf{c} \in \mathcal{C}$ and $\mathbf{r} \in \mathcal{R}$. These are used for embedding the hidden messages $\mathbf{m}$. Through the steganographic process, we obtain steg-images $\mathbf{s}$ of Figure 6-(c), which consist of the cover images $\mathbf{c}$ and messages $\mathbf{m}$. Figure 6-(d) and -(e) represent the images decrypted with correct passwords and with incorrect passwords respectively. By eye, it looks as if there is no difference betwen them. For a more scientific measurement, we calculated four different metrics to measure the similarity distances between $\mathbf{s}$, $\tilde{\mathbf{s}}$ and $\mathbf{h}$. Table 2 shows the similarity distances and their p-values between images, $\mathcal{D}_{type}(\mathbf{Y}, \tilde{\mathbf{s}})$ and $\{\mathcal{D}_{type}(\mathbf{Y}, \mathbf{h}^{(j)})\}_{j=1}^{200}$. As can be seen in the table, all p-values are larger than 0.05, a standard significance level used for hypothesis testing. That is, we can say that the stego-images are ϵ-bound in their difference from public images, i.e., $|(\mathbf{Y} - \tilde{\mathbf{s}}) - (\mathbf{Y} - \mathbf{h})| < \epsilon$.

Table 2: Similarity distance and its p-value between images, $\mathcal{D}_{type}(\mathbf{Y}, \mathbf{s})$ and $\{\mathcal{D}_{type}(\mathbf{Y}, \mathbf{h}^{(j)})\}_{j=1}^{200}$

	$I_{Einstein}$	I_{Rome}	I_{Lena}
$\mathcal{D}_{KLD}(\mathbf{Y}, \mathbf{s})$	0.036	0.056	0.041
(p-value)	(0.685)	(0.055)	(0.735)
$\mathcal{D}_{PSNR}(\mathbf{Y}, \mathbf{s})$	1.02	0.911	0.974
(p-value)	(0.935)	(0.485)	(0.55)
$\mathcal{D}_{RMSE}(\mathbf{Y}, \mathbf{s})$	24.38	31.31	27.05
(p-value)	(0.895)	(0.465)	(0.555)
$\mathcal{D}_{SSIM}(\mathbf{Y}, \mathbf{s})$	0.762	0.876	0.833
(p-value)	(0.85)	(0.41)	(0.52)

However, table 2 is an indirect metric for the honey property. In theory, honey encryption is defined by $|\tilde{\mathbf{s}} - \mathbf{h}| < \epsilon$. Therefore, we calculate the distances between $\tilde{\mathbf{s}}$ and $\mathbf{h}$s and present them in table 3. The average distance between $\tilde{\mathbf{s}}$ and $\mathbf{h}$s and are closely located at the mode of the distribution, which means that expected p-values are larger than the significance level 0.05. Therefore, we have shown that $|\tilde{\mathbf{s}} - \mathbf{h}| < \epsilon$ and conclude that practically our proposed VHE becomes honey steganography.

5.5 Security Analysis of Honey Steganography

As already referred to, Figure 3 shows three different frameworks for steganography:1) steganography without encryption modules, 2) steganography in an encrypted channel, and 3) our proposed Honey steganography. The outstanding performance of our proposed Honey steganography is shown by its high security compared to the other two approaches. From the attackers' point of view, we first define four func-

Method	Time Complexity for attack
Steganography without encryption modules	$T_{stego}(n)$
Steganography in an encrypted channel	$\|\Psi\|\{T_{generate}(n) + T_{decrypt}(n) + T_{rand}(n)\} + T_{stego}(n)$ $= \|\Psi\|\{T_{generate}(n) + T_{decrypt}(n)\} + \|\Psi\| T_{rand}(n) + T_{stego}(n)$
Our proposed honey steganography	$\|\Psi\|\{T_{generate}(n) + T_{decrypt}(n) + T_{stego}(n)\}$ $= \|\Psi\|\{T_{generate}(n) + T_{decrypt}(n)\} + \|\Psi\| T_{stego}(n)$

Table 4: Time comparison of various steganography for attack, n is the parameter for the size of image and $|\Psi|$ is the possible number of passwords or keys

Table 3: Similarity distances between images, $\{\mathcal{D}_{type}(\mathbf{s}, \mathbf{h}^{(j)})\}_{j=1}^{R}$ and $\{\mathcal{D}_{type}(\mathbf{h}^{(i)}, \mathbf{h}^{(j)})\}_{i,j=1,i\neq j}^{R}$

	$I_{Einstein}$	I_{Rome}	I_{Lena}
$\mathbf{E}[\mathcal{D}_{KLD}(\mathbf{s}, \mathbf{h})]$	0.052	0.060	0.053
(E[p-value])	(0.523)	(0.552)	(0.528)
$\mathbf{E}[\mathcal{D}_{PSNR}(\mathbf{s}, \mathbf{h})]$	0.915	0.866	0.897
(E[p-value])	(0.593)	(0.570)	(0.527)
$\mathbf{E}[\mathcal{D}_{RMSE}(\mathbf{s}, \mathbf{h})]$	30.99	34.70	32.29
(E[p-value])	(0.5924)	(0.568)	(0.526)
$\mathbf{E}[\mathcal{D}_{SSIM}(\mathbf{s}, \mathbf{h})]$	0.584	0.807	0.716
(E[p-value])	(0.597)	(0.557)	(0.530)

tions which count the elapsed times: (1) $T_{generate}(n)$: Consumed time when generating a key from a password of which size is $|\Psi|$, (2) $T_{decrypt}(n)$: Consumed time when decryption, (3) $T_{rand}(n)$: Consumed time when checking the randomness of the images, and (4) $T_{stego}(n)$: Consumed time when using stegoanalysis to extract the hidden messages. In this paper, let $|\Psi|$ be the number of possible passwords, password space. In general, $T_{generate}(n)$ and $T_{stego}(n)$ are much larger than $T_{decrypt}(n)$ and $T_{rand}(n)$ since often several hashing operations are used when generating a key from the password and steganalysis is known to be a time consuming operation. We simply assume that $T_{rand}(n) < T_{decrypt}(n) << T_{generate}(n) << T_{stego}(n)$. Given this assumption, the time taken for steganalysis to identify the stego-images and to extract the hidden messages are compared in table 4. As shown in the table, our proposed honey steganography needs $(|\Psi| - 1)T_{stego}(n)$ times more execution time because all decoded images should be processed by steganalysis because of their indistinguishability. In traditional steganographic algorithms, this is unnecessary because all decrypted and decoded images with incorrect passwords or keys are random except for the image decrypted with correct passwords or keys. Figure 7 displays the results of simulating the execution time complexities in table 4. As expected, an attack on our proposed approach requires much more execution time and therefore our approach is more secure.

6. IMPLEMENTATION ISSUES

6.1 The Effects of Different N

Since we are using a single image instead of N multiple public images, Alice and Bob should share the correct value of N. If Alice and Bob have different N, then Bob will not be able to obtain the correct data. In addition, we simulated the system with varying N. As can be seen in Figure 8, the decoded image becomes closer to the original fake one since the distribution is constructed with more fake images but the neighboured pixels' value is fixed.

(a) $N = 10$ (b) $N = 50$ (c) $N = 500$

Figure 8: The clarity with various N

6.2 Simplified DTE for Honey steganography

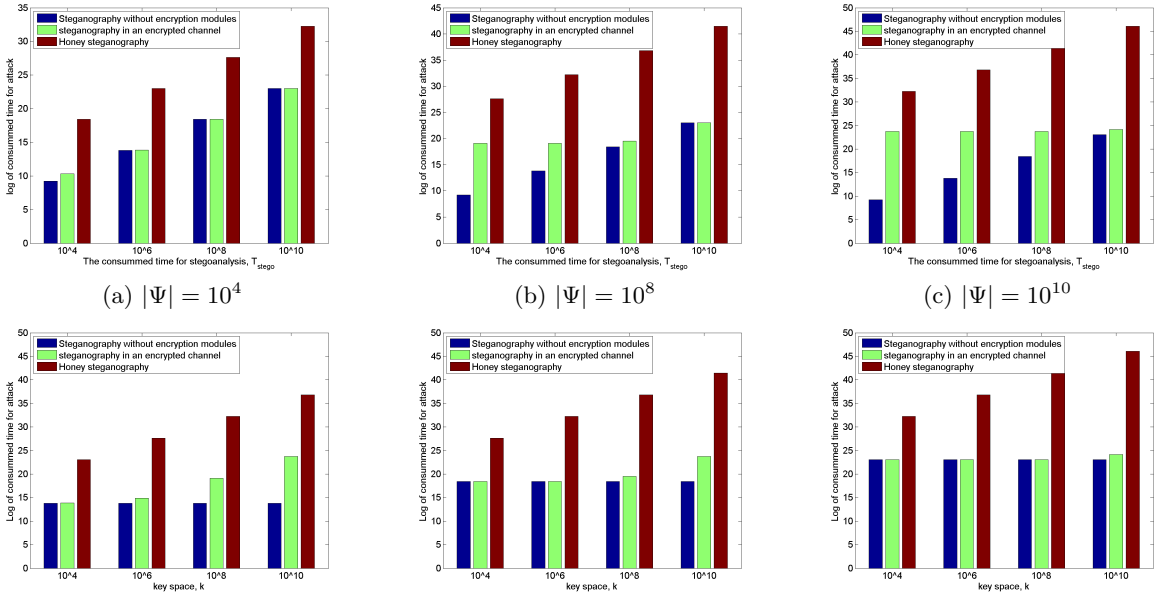
With equation (3) for a single fake image and (6) for multiple fake images, the curvature space $\mathbf{x}$ is assumed to be a d-th order Markovian model in order to allow for the smoothness and reality of the physical images. That is, $p(x_i|\mathbf{x}_{1:i-1}, \mathbf{Y})$ can be reduced to $p(x_i|\mathbf{x}_{ne(i)}, \mathbf{Y})$ causing much lighter computation. However, we can still simplify the modeling by assuming the independence between $x_i \perp x_j$ although $x_j \in ne(i)$ and the prior of x_i may follow a uniform distribution. Then, the target distribution using N multiple fake images becomes $p(\mathbf{x}|\mathbf{Y}, \theta) = \prod_{i=1}^{L} \left[\prod_{n=1}^{N} p(y_i^{(n)}|x_i, \theta)\right] \times p(x_i|\theta) = \prod_{i=1}^{L}\prod_{n=1}^{N} p(y_i^{(n)}|x_i, \theta)$. In this case, the fake images are the significant factors to consider when constructing the distribution.

7. CONCLUSIONS

It is known that conventional honey encryption only works in limited domains such as binary bit streams and integer sequences. However, there are many more complicated data types which have a synthetic or semantic structure including natural language based texts, images, videos. In this paper, we introduced a new variation of honey encryption which can be applied to such complicated data types. The proposed approach has been designed in a Bayesian framework and can be applied to create a new steganography scheme which requires a high time complexity for stegano-analysis. Using this new steganography scheme, this new steganography is more secure than any conventional steganography.

8. ACKNOWLEDGMENTS

This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning (No. NRF-2013R1A1A1012797). This work was also supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (No. 2014R1A1A1003707).



(a) $|\Psi| = 10^4$ (b) $|\Psi| = 10^8$ (c) $|\Psi| = 10^{10}$

(d) $T_{stego}(n) = 10^6$ (log scale) (e) $T_{stego}(n) = 10^8$ (log scale) (f) $T_{stego}(n) = 10^{10}$ (log scale)

Figure 7: Time comparison with varying a key size $|\Psi|$ and $T_{stego}(n)$

9. REFERENCES

[1] J. Barbier and S. Alt. Practical insecurity for effective steganalysis. In *Information Hiding, 10th International Workshop, IH 2008, Santa Barbara, CA, USA, May 19-21, 2008, Revised Selected Papers*, pages 195–208, 2008.

[2] J. Besag. Spatial interaction and the statistical analysis of lattice systems (with discussion). *Journal of Royal Statistical Society B*, 36:192–236, 1974.

[3] J. Besag. On the Staistical Analysis of Dirty Pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, 48(3):259–302, 1986.

[4] J. Bonneau. The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *Proceedings of Security and Privacy (SP)*. IEEE, 2012.

[5] J. Daemen and V. Rijmen. *The Design of Rijndael*. Springer-Verlag New York, Inc., 2002.

[6] D. Florencio and C. Herley. A large-scale study of web password habits. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 657–666, 2007.

[7] C. Herley, P. C. Oorschot, and A. S. Patrick. Passwords: If we're so smart, why are we still using them? In *Proceedings of Financial Cryptography and Data Security*, pages 230–237, 2009.

[8] A. Juels and T. Ristenpart. Honey encryption: Encryption beyond the brute-force barrier. *Security & Privacy, IEEE*, 12(4):59–62, 2014.

[9] A. Juels and T. Ristenpart. Honey encryption: Security beyond the brute-force bound. In *Advances in Cryptology – EUROCRYPT*, pages 293–310, 2014.

[10] B. Kaliski. PKCS #5: Password-Based Cryptography Specification Version 2.0, 2000.

[11] H. Kim and J. H. Huh. PIN selection policies: Are they really effective? *Computers & Security*, 31(4):484–496, 2012.

[12] H. Rue and L. Held. *Gaussian Markov Random Field: Theory and Applications*. Chapman & Hall/CRC, 2005.

[13] A. Skillen and M. Mannan. On Implementing Deniable Storage Encryption for Mobile Devices. In *Proceedings of the 20th Annual Network & Distributed System Security Symposium*, NDSS '13, February 2013.

[14] Z. Wang and A. C. Bovik. A universal image quality index. *Signal Proc. Lett., IEEE*, 9(3):81–84, 2002.

[15] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[16] J. W. Yoon. Statistical denoising scheme for single molecule fluorescence microscopic images. *Biomedical Signal Processing and Control*, 10(0):11 – 20, 2014.

APPENDIX

A. COLLAPSING MULTIPLE NORMAL DISTRIBUTIONS

If we have two normal distributions $\mathcal{N}(x;\mu_1,\sigma_1^2)$ and $\mathcal{N}(x;\mu_2,\sigma_2^2)$, the collapsed normal distribution is as follows: $\mathcal{N}(x;\mu,\sigma^2) = \mathcal{N}(x;\mu_1,\sigma_1^2)\mathcal{N}(x;\mu_2,\sigma_2^2)$ where $\sigma = \sqrt{(1/\sigma_1^2 + 1/\sigma_2^2)}$ and $\mu = \sigma^2(\mu_1/\sigma_1^2 + \mu_2/\sigma_2^2)$.

For multiple normal distributions, we can generalize the product of the multiple normal distributions: $\mathcal{N}(x;\mu,\sigma^2) = \prod_{n=1}^{N}\mathcal{N}(x;\mu_n,\sigma_n^2)$ where $\sigma = \sqrt{\left(\sum_{n=1}^{N}\frac{1}{\sigma_n^2}\right)^{-1}}$ and $\mu = \sigma^2\sum_{n=1}^{N}\frac{\mu_n}{\sigma_n^2}$.