A Novel Random Email-Based Steganography

Tohari Ahmad, Melvin S. Z. Marbun, Hudan Studiawan, Waskitho Wibisono, and Royyana M. Ijtihadie

***Abstract*—Sharing information over public network is susceptible. It is possible that the information being sent is intercepted by an adversary. One possible solution to alleviate this problem is by using email-based steganography which does not produce any noise. This makes it difficult for an adversary to recognize the secret data. Different from previous methods, in this paper, we propose to generate relatively random email addresses to be the keys to reconstruct the secret which has been embedded into the email text. The experimental results show that this method has a good performance.**

***Index Terms*—Steganography, email, secret message, data security, confidentiality.**

I. Introduction

In this digital era, sharing information has been a main need for many people. Furthermore, there are many tools can be used for this purpose, such as social media and email. Nevertheless, security is still a concern. It is possible that data being sent to another party is intercepted by illegitimate users. On one hand, this can have a big impact, especially, for classified data which can be used for compromising other data/systems. On the other hand, there are many tools for committing such illegitimate activities available on the internet.

Securing transferred data can be done by using a private network which requires a relatively high cost. This may not be efficient to build this network, in terms of cost and time, if the data transmission is only rarely performed. A possible solution is by sending protected data over a public network. In this case, protection is only given to the specified data. It does not need to protect common data which is sharable to the public.

There are some protection methods have been known and implemented. Those are cryptography, watermarking and steganography. The first works by modifying the original data while the last two works by hiding the data [1]. In particular, cryptography transforms the original data into relatively random data. It is hard to recover the original data just based on its transform format. Watermarking and steganography embed the secret data into cover data so that it is hard to identify the secret just from the cover. An advantage of watermarking or steganography is that it may not attract people attention since the embedded data (stego data) is relatively similar to the cover. This characteristic is different from cryptography where the encrypted data has totally different format from its corresponding unencrypted/plain data.

In addition, watermarking may not have security property because it is relatively easy to destroy if the embedded data is modified [2]. This has made it useful for certain purposes, such as digital right management, and tracing transaction. Slightly different from it, steganography uses a common medium to carry the secret data, for example, text, audio and video. In general, there are some factors should be considered in designing a steganographic method. These include security of secret data and capacity of secret data can be embedded into the cover. In some cases, steganography may not provide protection to the secret data [2]. In order to minimize this problem, steganography is often implemented along with cryptography. That is, the secret data is encrypted before it is embedded into the cover [3]. In practice, similar to cryptography, steganography also requires a key, called a stego key, to embed and to extract the secret data.

In [4], [5], Satir and Isik propose to use an email to be the cover of the secret data. It can be assumed that an email is very popular that people use it as one of the main communication tools. A characteristic exploited in this method is the fact that an email can be sent to many recipients in a single process. There may not be a proper method to validate whether the email address of each recipient is valid but sending an email to the respective address. In those papers [4], [5], email addresses are used to hide the stego key, regardless they are valid addresses or not. These addresses are generated based on the single secret data. However, its relatively high number may attract people attention, in case the email is intercepted. Furthermore, the generated email addresses are readable which make it possible to send the email to real but illegitimate users.

In this paper, we propose to solve those problems. It is also intended to avoid curiosity of other users, in case the email is disclose. The rest of the paper is structured as follows. Section II provides the related works. Section III and Section IV describe the proposed method and the experimental result. Finally, the conclusion is drawn in Section V.

II. Related Works

Generally speaking, there are some differences between cryptography and steganography, even though both are used for the same purpose: hiding the secret data. Those differences include:

- Steganography requires medium to hide the secret data, while cryptography does not
- In general, it is more difficult for people to differentiate data before and after being transformed by a steganographic method than that of cryptographic method.

The process of the steganography can be depicted in Fig. 1.

Manuscript received October 15, 2013; revised December 16, 2013.

Tohari Ahmad, Melvin S. Z. Marbun, Hudan Studiawan, Waskitho Wibisono, and Royyana M. Ijtihadie are with Department of Informatics, Institut Teknologi Sepuluh Nopember (ITS), Kampus ITS Surabaya, Indonesia (email: tohari@if.its.ac.id).

In order to increase the capacity of the secret data to be embedded into the cover, the secret should be compressed before being processed. In this case, the compression method must be able to recover the secret exactly same as the original, called lossless. It means that there is no difference between before and after the compression-decompression [6]. This mainly uses huffman coding by developing a tree which is considered efficient to use.

Secret data + Cover data = Stego data

Fig. 1. Process of steganography.

Nowadays, there are many steganographic methods introduced which employ public medium to be the cover. These can be an image, text, audio or video which is used in the internet forum, chat, email, etc. Overall, steganography can be classified into three categories: spatial, frequency and adaptive [7] where each of them has different strenght and weakness. In the spatial domain, particularly in an image, steganography can be done by manipulating LSB; in the frequency domain, it can be performed by either discrete cosinus transforms (DCT) [8], Fourier transforms or discrete wavelet transforms (DWT) [9].

In a chat application, Wang et al. [10] employ emoticons (emotional icons) to hide a secret message. This is intended to improve the performance of other chat-based steganography. The proposed method is developed based on the assumption that the use of emoticons in chatting is high, where many recent applications using it. It is claimed that this method is able to raise up the capacity of the secret data to be embedded as well as to provide an easy to use application. This method requires both sender and receiver have an exactly same emoticon table containing some emoticon sets. In this case, the order of the emoticon affects the meaning of the secret message. Therefore, there must be a syncronization process before the communication begins. Practically, this method requires the users to input many emoticons in order to send the secret data. This may have an effect on the user convenience. Moreover, this can also attract attention of other users who see this chat channel. Other research has been carried out which employ varied communication medium, such as in [11], [12].

Still in social networking, by manipulating text, Ren et al. [13] introduce a method which can hide a secret data in a short text. It is designed to have a certain difficulty level such that it is hard to disclose the secret. There are two constrains in developing this design. First, it should be secure. It means that there is no party be able to retrieve the secret without having the concealment knowledge. Second, it should have good usability and efficency. It means that the proposed method not only implementable and easy to use but also generates less overhead (cost). The evaluation, which are done by measuring the computationally security level, and concealment rate along with its ease to use level, give a good result.

Hiding the secret in the text is also investigated by Desoky [14]. He proposes Listega, a method to hide data which exploits the popularity of list of items, such as song, food, drink and car. Here, the secret data is encoded and correlated with items. This produces a list of items which is to be the cover. In order to further camouflage the secret, the list of items can be mixed with plain items. This arrangement can be random which makes it more difficult for the adversary to find the secret. Moreover, this may also reduce people attention to the text because it looks like ordinary list of items. The experimental result shows that the proposed method has a good performance. It is depicted by its superior bitrate to other contemporary linguistic steganogaphy methods. Nevertheless, the extraction procedure may not be so complex and the capacity of the secret can be embedded into the cover depends on the number of items used for creating the list (cover).

In addition to that list-based steganography, Desoky [15], introduces note-based steganography, called notestega, in 2009. Similar to the previous method [14], this method does not generate detectable noises or employs noises in order to hide the secret. He proposes to conceal the data by exploiting the differences between the result of automatic notetaking and human notes. There are three stages in this technique. Firstly, the same input is used to produce variouse notes. Secondly, certain notes are selected based on the specific protocol. Lastly, text in the non-selected notes resulted from the second stages replaces text in the selected notes. It is claimed that this method results to a good performance and is able to provide enough space to hide secret data. However, notestega must be carefully implemented. This is to prevent the adversary from identifying the suspicous pattern generated in that third stage.

In 2012, Satir and Isik [4] propose an LZW compression-based text steganographic method which consider both the capacity of the secret data and security factors. The first purpose is achieved by implementing the LZW method. Moreover, LZW is popular and has good performance. The second purpose is by designing combinatorics-based coding and stego keys. The secret data is embedded into text in the email. The experimental result presents that the capacity can be up to 7.042%, particularly if the secret message consists of 300 characters. Comparing this capacity size to other methods depict that it is superior.

The proposed method in [4] requires to define the email domains before the process starts; while the respective user name is generated partially based on the secret and cover data. The email addresses generated are put in the ”cc” of the email. Therefore, those email addresses may not be valid. These are used in the extraction process to obtain the secret from the cover. These generated email addresses and the cover themselves are combined to be the stego data.

In the next research, Satir and Isik [5] make further improvement of their previous method [4]. Different from that, Huffman-based compression is used instead of LZW. It is intended to have bigger capacity so that more secret can be embedded into the cover. In addition, Huffman method is chosen because it is predicted that, in this case, Huffman method is more appropriate to increase the security level. This is because the process of obtaining the secret is harder. Furthermore, it is also believed that Huffman-based compression method has a good compression ratio.

III. Proposed Method

Our proposed method is developed based on those methods presented in Section II, particularly [4], [5]. This method still

uses Huffman coding because it is considered to have a good performance, especially in this design. The difference is on the process of email address generation. This includes adding variables and replacing some keys.

A. Embedding Phase

The embedding phase consists of some steps including setting up some variables used in the process. Steps 1 and 2 are similar to those in [4], [5], which can be described as follows.

Step 1. Firstly, it needs to specify the email domain will be used in the stego keys. This can be, for example: A = {gmail.com, yahoo.com, hotmail.com, msn.com, aol.com, hotmail.co.uk, yahoo.co.uk, live.com}. This set of email domain is assigned to the binary number as shown in Fig. 2, such that each of them has a unique number.

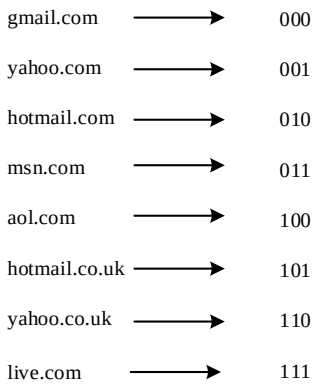

Fig. 2. Mapping email domain.

Let *a* be a set of characters in the secret message *S*, and *b* be a set of characters in the cover text *T*. It is to find $a = b$ and to form a vector $\overrightarrow{\Delta D}$ whose elements *c* are the distance between elements *b*. As in [4], this can be illustrated in Fig. 3.

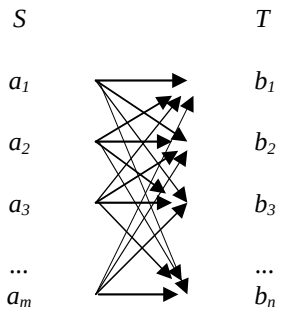

Fig. 3. Character mapping.

Iteratively, the elements of $\overrightarrow{\Delta D}$ are determined based on the assumption that: $a_x = b_y$, where $x \in \{1, 2, \dots, m\}$ and $y \in \{1, 2, \dots, n\}$. In general, if $a_1 = b_1$ then $a_1 - b_1$ is 0. Equivalently, if $a_1 = b_n$ then $a_1 - b_n$ is also 0. Here, the value of $\overrightarrow{\Delta D}$ depends on the index of *b*. For example, if $a_1 = b_3$ then 3 is taken.

This also works for a_2 that if $a_2 = b_3$ then $a_2 - b_3$is 0. The value of $\overrightarrow{\Delta D}$ is determined by the distance between the index of *b* and that of previous *b*. For example, let $a_2 = b_5$. This index of *b* is substracted by the previous index of *b*. So, 5-3=2. The similar process is performed to all characters in *S* and *T*. Therefore, at the end, we have the value of $\overrightarrow{\Delta D}$.

Step 2. After $\overrightarrow{\Delta D}$ has been obtained, it needs to check the number of elements of D. This can be presented as in Eq. 1 and Eq. 2.

$$\text{if } D > 26 \begin{cases} \vec{E} = D/26 \\ \vec{R} = D \ mod \ 26 \end{cases} \quad (1)$$

$$\text{if } D \leq 26 \begin{cases} \vec{E} = 0 \\ \vec{R} = D \end{cases} \quad (2)$$

Step 3. The Huffman-based compression method is implemented to $\vec{R}$ which results to binary values. The value of $\vec{R}'$ is generated by combining the result of compression of $\vec{R}$ into a bitstream. Information of the lenght of each element $\vec{R}'$ is stored in a list of keys (K_3). The lenght of bitstream is checked whether it is multiples of 12. If it is not, then append 0 to the end of bitstream until multiples of 12 is achieved.

Step 4. The resulted bitstream $\vec{R}$'is divided into groups of 12 bits. In each group, the first 9 bits (written as G_1) is used for generating the user name of the email (the part before '@' in an email address). This follows the Eq. 3, where $(G_m)_n$ means converting binary number G_m into *n*-based numbers.

$$\left.\begin{array}{c} v = \frac{(G_1)_{10}}{26} + 13 \\ w = (G_1)_{10} mod \ 26 + 13 \\ x = \frac{(G_1)_{10}}{26} \\ y = (G_1)_{10} mod \ 26 \end{array}\right\} \quad (3)$$

Each variable (*v, w, x, y*) is mapped to the Latin square. Different from that in [4] [5], this Latin square contains 52 different characters which consists of both upper and lowercase. Detail of the mapping procedure is: the first group is to row 1, the next group is to row 2, and so on. This is done for all groups.

Similar to [4], the last 3 bits of $\vec{R}$' (written as G_2) are used to generate email domains of the email (the part after '@' in an email address), such as: gmail.com and yahoo.com. This is to get *z* (see Eq. 4) which is then mapped to A.

$$z = (G_2)_{10} \quad (4)$$

Step 5. This step is to obtain random words to be part of the user name. It is done by randomizing integer between 4 and 7 whose result is store into a variable *word_length.* This process is iteratively performed *i* times, where *i* is the number of groups in step 4.

Mapping the integer resulted from *j* iteration to the Latin square is carried out to randomize integer between 1 and 52; where, in this case, *j* is the value of the variable *word_length.* The purpose of generating these characters is to make the user name of the email address harder to crack; even though they are actually meaningless to the secret.

Step 6. This step is to construct K_2 according to Eq. 5.

$$K_2 = \{gemail_1, gemail_2, \dots, gemail_n\} \quad (5)$$

where $gemail_n$ is the generated email adress. Each group of 12 bits in step 4 produces one $gemail_n$, while $gemail_n$ itself is constructed according to Eq. 6.

$$gemail_n = mapping_v + mapping_w + mapping_r + mapping_x + mapping_y + '@' + mapping_z \quad (6)$$

where $mapping_v$, $mapping_w$, $mapping_x$ $mapping_y$, and $mapping_z$ are obtained from step 4 while $mapping_r$ are from step 5. It can be inferred that each generated email address has a unique user name and is likely to be an inexist address. This is appropriate to its creation purpose that they should be different from the real one. An example of generated email is shown in Fig. 4, where the process outputs cyZRQCMU@gmail.com.

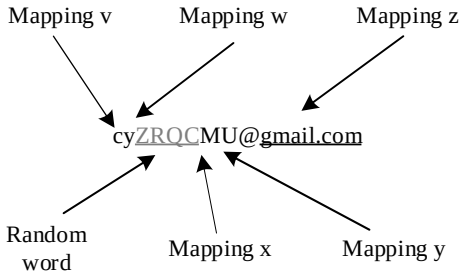

Fig. 4. An email address construction.

As shown in Fig. 4, an email address consists of 6 parts, namely: 4 characters generated from mapping *v*, mapping *w*, mapping *x* and mapping *y*; a random word; and an email domain generated from mapping *z*. The stego key is stored in the five mapping results (mapping *v*, *w*, *x*, *y* and *z*); while the random word is only for camouflage. Therefore, the first two and the last two characters of the user name as well as the email domain are used in extracting the secret data.

Step 7. This is to develop stego cover which is combination of K_2 and *T*. Overall, this proposed embedding phase, which is developed based on that of [4], [5], is presented in Fig. 5.

B. Extracting Phase

There are some variables must be held in order to extract the secret data. Those are:

- Key 1: the value of $\vec{E}$
- Key 2: the stego key
- Key 3: the number of bits of each Huffman codeword
- A: the list of email domains
- Latin square table and Huffman coding tree

In this paper, it is assumed that the extractor has all those variables. Since the extraction process is actually the reverse of the embedding process, both are equivalent. For example, based on the email structure depicted in Fig. 3, each character of user name is extracted by using the respective table, i.e., variables *v*, *w*, *x*, *y* are processed in Latin square table, and variable *z* is in the email domain table. This extraction process can be described as follows.

Step 1. This is to have value of *v*, *w*, *x*, *y* and *z* of each email adress. Those of the first email are mapped into the first row of the Latin square table, those of the second email is into the second row, and so on. In order to obtain the value of *z*, the email domain address is mapped into variable *A*.

Step 2. We must check whether each email meets the requirements: $x = v - 13, y = w - 13$. The value of G_1 is obtained according to Eq. 7, while G_2 is by converting the index of *z* to binary. Combination of G_1 and G_2 produces a 12 bit number. This process is performed for all email addresses.

$$G_1 = ((26 \times x) + y)_2 \quad (7)$$

Step 3. All 12 bit numbers generated from step 2 are combined into a bitstream. We use the information stored in K_3 to divide this bitstream according to its codeword lenght. In case there are some '0' at the end which can not be reached by K_3, we can ignore them. These '0' are appended during the embedding phase in order to make its lenght multiple of 12. Once the codewords have been formed, change each codeword to a number according to the Huffman coding tree. At this stage, we already have the value of $\vec{R}$.

Step 4. For each correlating index, $\vec{R}$ value is added to $\vec{E}$ value. For example, $\vec{R}$ whose index is 0, is added to $\vec{E}$ whose index is also 0, and so on. All of these values develop $\overrightarrow{\Delta D}$.

Step 5. Based on the $\overrightarrow{\Delta D}$, we can reconstruct the secret (*S*) from the cover text (*T*). Overall the extracting process can be described in Fig. 6, which is an improvement of that of [4] and [5].

IV. Experimental Results

The proposed method is evaluated by using a secret data whose lenght is 87 characters and some cover data whose lenght is varied: 547, 1200, 1802 and 2433 characters. Those covers are respectively symbolized as T_1, T_2, T_3 and T_4.

The experimental result shows that the length of the cover affects the successfulness of the embedding process. This is because relatively short cover text may not be able to represent the secret. For example, T_1 can not be used to hide the secret. It can only accomodate 36 characters in the secret.

Increasing the lenght of the cover, as in T_2, is able to increase the number of characters in the secret to be embedded; nevertheless, there are still some characters can not be included. There are only 56 characters out of 87 can be covered.

Different from the result of those first two cover data, T_3 successfully hides the secret. There are 29 email addresses generated based on those secret and cover data. Likewise, T_4 is also able to hide the secret by generating 29 email addresses, similar to that of T_3.

Practically, not only the lenght of the cover data determines the succesfulness of the embedding process, but also variation of the characters. Longer the cover data actually increases the possiblity of providing more varied characters. Therefore, successfulness of the embedding process depends on the appropriate selection of the cover data.

The result of the experiment can be summarized in Table I. It shows that for certain numbers of cover data, the lenght of bitstream and number of generated email addresses are same.

The proposed method has been able to generate email addresses whose user name is relatively random, different from that of [4], [5]. There is strenght and weakness of this condition. Random user names may attract more other user attention, in case other users read that email; even though, as far as we know, there has not been research about this relation. In addition, non-random user names are likely same with valid email addresses which really exist. In this case, sending an

email to those addresses can reduce the security because the email has become public. This may not happen with random email addresses which may not really exist.

Get S
Get T
For each character in T
    Calculate $\overrightarrow{\Delta D}$
End for
For each c in $\overrightarrow{\Delta D}$
    If c > 26 then
      e = int (c/26)
      r = c mod 26
    else
      e = 0
      r = 0
Generate $\vec{E}$ based on e
Generate $\vec{R}$ based on r
Generate $\overrightarrow{R'}$ using Huffman Coding
Construct K_3 based on n of each element $\overrightarrow{R'}$
Bit stream = $(\overrightarrow{R'})_2$
Generate 12 bit groups by separating Bit stream
For each 12 bit groups in Bit stream
    Randomize int between 4 and 7
    i=0
    For i in range randomize int
      Randomize another int between 1 and 52
          Map the another random int to first row in Latin Square Table
      Concat every mapped character into a word
    Get G_1
    $v = (G_1)_{10}/26 + 13$
    $w = (G_1)_{10} \bmod 26 + 13$
    $x = (G_1)_{10}/26$
    $y = (G_1)_{10} \bmod 26$
    Generate letters according to r v, w, x and y in Latin Square
    Get G_2
    $z = (G_2)_{10}$
    Generate a stego key based on the value of z in A
    Construct K_2 according to : mapping v + mapping w + randomize word + mapping x + mapping y + '@' + mapping z
End for
Contruct stego cover by combining K_2 and T

Fig. 5. Modified pseudo code of embedding phase of [4].

TABLE I: EXPERIMENTAL RESULTS

cover	Length of cover (characters)	Result	Length of bit stream (bit)	Number of generated email addresses
T1	547	Fail	-	-
T2	1200	Fail	-	-
T3	1802	success	348	29
T4	2433	success	348	29

V. CONCLUSION

The paper has proposed a text-based steganography which is appropriate to use in an email environment. There are some improvements of the previous method, such as the randomity of the generated email addresses. In this case, random generated email addresses may increase the security level.

The length and variation of the cover text determine the successfulness of the embedding process. A short and non-varied cover text is likely to cause the embedding proses fails. So, the embedding process may be run several times by changing the cover.

Get stego cover
For each k in K_2
    Get v by mapping first character to Latin Square
    Get w by mapping second character to Latin Square
    Get x by mapping last two character before '@' to Latin Square
    Get y by mapping last character before '@' to Latin Square
    Get z by map the corresponded email address to A
    If v != x + 13 && w != y+13
      Print "Wrong Email "
      Break
    $G_1 = (x.26)_2$
    $G_2 = (z)_2$
    12 bit groups = G_1+G_2
    Get Bit stream by concatenated each 12 bit groups
End for
Generate codewords of $\vec{R}$ by separating bit stream using K_3
Get $\vec{R}$ by mapping codewords using Shared Huffman Tree
For each r in $\vec{R}$
    $\overrightarrow{\Delta D}[i] = r.26 + \overrightarrow{E[i]}$
End for
For each c in $\overrightarrow{\Delta D}$
    a = Text[c]
    Generate S based on a
End for

Fig. 6. Modified pseudocode of extracting phase of [4].

REFERENCES

[1] C. Chang and T. Kieu, "A reversible data hiding scheme using complementary embedding strategy," *Information Science*, vol. 180, no. 16, pp. 3045-3058, 2010.

[2] A. Gutub and M. Fattani, "A novel Arabic text steganography method using letter points and extensions," presented at WASET International Conference on Computer, Information and Systems Science and Engineering, Vienna, 2007.

[3] J. Condell, K. Curran, and P. McKevitt, "Securing information content using new encryption method and Steganography," in *Proc. Third International Conference on Digital Information Management*, 2008.

[4] E. Satir and H. Isik, "A compression-based text steganography method," *The Journal of Systems and Software Science Direct*, vol. 85, issue 10, pp. 2385-2394, 2012.

[5] E. Satir and H. Isik, "A Huffman Compression based Text Steganography Method," *Multimedia Tools Appl*, September 2012.

[6] D. Salomon and G. Motta, Handbook of Data Compression, London, England: Springer, 2010.

[7] A. Cheddad, J. Condell, K. Curran, and P. M. Kevit, "Digital image steganography: Survey and analysis of current methods," *Signal Processing*, vol. 90, pp. 727–752, 2010.

[8] N. Ahmed, "Discrete Cosine Transform," *IEEE Transactions on Computers*, vol. C-23, pp. 90-93, 1974.

[9] M. J. Shensa, "The discrete wavelet transform: wedding the a trous and Mallat algorithms," *IEEE Transactions on Signal Processing*, vol. 40, pp. 2464-2482, 1992.

[10] Z. Wang, T. Kieu, C. Chang, and M. Li, "Emoticon-based text steganography," in *Proc. 2009 Asia-Pacific Conference on Computational*, Wuhan, China, 2009.

[11] L. Por, K. Wong, and K. Chee, "UniSpaCh: a text based data hiding method using unicode space characters," *Journal of Systems and Software*, vol. 85, issue 5, pp. 1075-1082, 2012.

[12] M. Topkara, U. Topkara, and M. Atallah, "Information hiding through errors: a confusing approach," in *Proc. SPIE International Conference*

on Security,Steganography, and Watermarking of Multimedia Contents, San Jose, CA, 2007.
[13] W. Ren, L. Yuliang, and Z. Junge, "Provably Secure Information Hiding via Short Text in Social Networking Tools," *Tsinghua Science and Technology*, vol. 17, no. 3, pp. 225-231, 2012.
[14] A. Desoky, "Listega: list-based steganography methodology," *International Journal of Information Security*, vol. 8, no. 4, pp. 247-261, 2009.
[15] A. Desoky, "Notestega: Notes-based Steganography Methodology," *Information Security Journal: A Global Perspective*, vol. 18, no. 4, pp. 178-193, 2009

Tohari Ahmad has obtained his bachelor degree, master degree and PhD from ITS, Monash University and RMIT University, respectively. All are in computer science and information technology. His research interest is in biometric security and information security.

Melvin S. Z. Marbun is a final year student at Department of Informatics, ITS. His research interest is in network security.

Hudan Studiawan has graduated from ITS for both bachelor degree and master degree. His research interest is in data compression and server cluster. He is also Certified Ethical Hacker (CEH) and become network security enthusiastic. Hudan is now lecturer in Department of Informatics, ITS.