Journal of Information Technology and Applications
Vol. 3, No. 1, pp. 13-24, 2008

Data Hiding in Emails and Applications Using Unused ASCII Control Codes

I-Shi Lee*
National Chiao Tung University / Department of Computer Science
Technology and Science Institute of Northern Taiwan / Department of Management Information
E-mail: gis87809@gmail.com

Wen-Hsiang Tsai
National Chiao Tung University / Department of Computer Science
Asia University / Department of Computer Science and Information Engineering
E-mail: whtsai@cis.nctu.edu.tw

Abstract

A new technique for hiding data in emails via Outlook Express and IE is proposed. Secret data are encoded by special ASCII control codes and embedded into cover emails by inserting the data into email text line ends. These special codes were found by systematically testing all the ASCII codes of various email server software systems and standards. The codes are invisible to the user when displayed, achieving the effect of steganography. The proposed data encoding technique is a combination of five coding rules. The inserted codes do not change the meaning of the sentences in the cover email, neither cause any noticeable difference to the reader. Furthermore, hidden data can be completely extracted from a stego-email to recover the original email text. Also, two applications of the proposed data hiding technique, namely covert communication via emails and authentication of emails, are described. In the first application, security is enhanced by the use of a secret key, and in the second, an authentication signal is generated from the cover email for email fidelity checking. Good experimental results show the feasibility of the proposed method.

Keyword: data hiding, ASCII control codes, email, secret data, covert communication, email authentication.

1. Introduction

With the advance of network technology, a great deal of information is transmitted on the Internet. However, some users might use the Internet to collect, copy, or intercept other people's data illegally. Cryptography is generally adopted to protect important data from being tampered with. Another solution to this problem is to use a data hiding technique, which embeds crucial data into given carriers, making the embedded information imperceptible to viewers. The given carrier may be any kind of multimedia like text, video, audio, or image [1-3]. In the rest of the paper, the carrier into which information is to be embedded will be referred to as *cover carrier*, and the result of the embedding as *stego-carrier*. In this study, we deal with emails, so a *stego-email* is the result of information embedding in a *cover email*.

In contrast with other multimedia, digital texts contain less redundant information for embedding data. Most data hiding methods for digital text documents try to encode information directly *into the text itself* or *into the text format*. One way of into-text hiding is using the natural redundancy of languages, and one way of into-format hiding is adjusting inter-word or inter-line spaces [3]. On the other hand, from the steganographic point of view, digital text documents can be classified into two types: hard-copy and soft-copy [2]. A hard-copy text may be treated as a binary image resulting from scanning a text document, while a soft-copy text may be regarded as an American Standard Code for Information Interchange (ASCII) text that can be edited by text editing software like Notepad.

For a hard-copy text, which is interpreted as a highly-structured image, information can be embedded into the layout or format of the image. Maxemchuk et al. [4-5] presented text-based steganographic methods which use the distances between consecutive lines of texts or between consecutive words to hide information. If the space between two lines is smaller than a threshold, a "0" is represented; otherwise, a "1."

In contrast with hard-copy texts and other digital media, soft-copy texts are more difficult to hide data due to the lack of redundant information. Even a slight modification, like rewriting a letter, may be noticed by a reader. However, huge amounts of text dealt with daily on the Internet are essentially soft-copy texts. Thus, the protection of the digital rights of this type of text document is becoming more and more important.

Bender et al. [2] proposed the use of infrequent additional spaces to form secret data and transmit them in soft-copy texts, including inter-sentence

* Corresponding author.

spacing, end-of-line spacing, and inter-word spacing. For example, one space between words is taken to represent a "0" and two spaces a "1." Wayner [6] proposed a method to use the context-free grammar to create secret text messages in cover files for covert communication; the secret message is not embedded in the cover file directly. And a receiver extracts the hidden message by parsing. A constraint is that the cover text must be a meaningful message, otherwise a reader will distrust it.

Cantrell and Dampier [7] proposed to embed data into unused spaces in file headers. These spaces are invisible to usual users because they are disregarded when the files are opened. The spaces can be seen when examined at the byte level, but few users would do so. Johnson et al. [8] proposed another way to embed information in unused spaces that are imperceptible to an observer, which is based on the fact that operating systems usually allocate more space than necessary, leaving some unused space to hide information in. A third method is to create a hidden partition in a file system to embed information. The partition is not viewed normally. This concept has been expanded in a steganographic file system [9]. If a user knows the file name and the password, access to the file will be granted; otherwise, no evidence of the file will be revealed in the system of the hidden files.

Characteristics inherent in network protocols may also be taken advantage of to hide information [10]. For example, TCP/IP packets can be used to transmit secret messages across the Internet by embedding unused spaces in the packet header. Finally, Chang and Tsai [11] proposed a special space encoding to embed copyright information into the HTML text content. The bit "1" is encoded by inserting a so-called *pseudo-space string* " " before a real space, while the bit "0" is represented by a normal space between two words or sentences.

It is impossible for a single administrator to check all data on networks daily because of the huge amount of data. For this reason, email, a kind of digital text document, is very suitable for use as a cover carrier to hide secret information. Nowadays, most users transmit and receive emails either by an application software like Outlook Express of Microsoft or by a browser like Internet Explorer (IE). The type of email handled by the latter is called *web mail*.

In this paper, a new technique for data hiding in emails via Outlook Express and IE under the operating system of the traditional Chinese version of Microsoft Windows XP, service pack 2, 2002 is proposed. The idea is based on the use of unused ASCII codes. Secret data are encoded by special ASCII control codes and embedded into cover emails by inserting the data into the text line ends in the body of a given email. These ASCII control codes, when displayed by both Outlook Express and IE, are invisible to the user, achieving an effect of steganography. Such invisible ASCII control codes were found in this study by systematically testing all the ASCII codes on various email server software systems and standards. The proposed data encoding technique is a combination of five coding rules found in this study, which insert special ASCII control codes into different places in email texts. The inserted codes do not change the meanings of the sentences in the cover email, neither do they cause any noticeable difference to the reader. Furthermore, hidden data can be completely extracted from a stego-email to recover the original email text content. Also, two applications of the proposed data hiding technique, namely covert communication via emails and authentication of emails, are described. In the first application, security is enhanced by the use of a secret key, and in the second, an authentication signal is generated from the cover email for email fidelity checking.

In the remainder of this paper, some properties of ASCII control codes and some email systems useful for this study are described in Sections 2 and 3, respectively. The techniques proposed for encoding secret data by ASCII control codes and embedding them into emails are described in Section 4. In Section 5, an algorithm which implements the proposed data hiding method for covert communication is described. In Section 6, the proposed data recovery technique and a corresponding algorithm are given. In Section 7, an algorithm for the application of email authentication is described. Some experimental results are given in Section 8, followed by conclusions and discussions in Section 9.

2. Properties of ASCII Control Codes

ASCII codes, usually expressed as hexadecimal numbers, are very commonly used to represent text for information interchange on computers. Some of the ASCII codes, namely from 00 through 1F, are used as *control codes*, as listed in Table 1. They were originally designed to control computer peripheral devices like printers, tape drivers, teletypes, etc. But now, except for the codes for text display control, such as 0A with the meaning of *line feed* and 08 with the meaning of *backspace*, they are rarely used for their original purpose because of the rapid development of new peripheral hardware technology. Besides, under certain software environments some of the control codes, when displayed by a text editing program or a browser on monitors, are *invisible*, and some others are shown as *spaces* just like the original ASCII space code 20. These two types of ASCII codes may be utilized to increase secret data encoding variability in the data hiding process. For

the convenience of reference, it said that a *null space* is displayed by the former type, in contrast with the *white space* displayed by the latter type.

Table 1: ASCII control codes and description

Dec	Hex	Char	Description
0	0	NUL	null character
1	1	SOH	start of header
2	2	STX	start of text
3	3	ETX	end of text
4	4	EOT	end of transmission
5	5	ENQ	enquiry
6	6	ACK	acknowledge
7	7	BEL	bell (ring)
8	8	BS	backspace
9	9	HT	horizontal tab
10	A	LF	line feed
11	B	VT	vertical tab
12	C	FF	form feed
13	D	CR	carriage return
14	E	SO	shift out
15	F	SI	shift in
16	10	DLE	data link escape
17	11	DC1	device control 1
18	12	DC2	device control 2
19	13	DC3	device control 3
20	14	DC4	device control 4
21	15	NAK	negative acknowledge
22	16	SYN	synchronize
23	17	ETB	end transmission block
24	18	CAN	cancel
25	19	EM	end of medium
26	1A	SUB	substitute
27	1B	ESC	escape
28	1C	FS	file separator
29	1D	GS	group separator
30	1E	RS	record separator
31	1F	US	unit separator

On the other hand, as computer technology is spreading, many coding standards have been developed to facilitate the expression of non-English alphabets. These alphabet coding standards, such as the Unicode and the Big 5, all include the ASCII codes as the kernel set. For example, the popular Unicode standard, UTF-8, equates exactly to the ASCII codes for code values below 128. Therefore, the features of the ASCII control codes that can be used for embedding secret data in text documents is still preserved in various coding standards.

In this study, the white-space and null-space codes are used to embed data in text documents of the Unicode UTF-8 format without causing noticeable artifacts under the popular software environments of Outlook Express, IE, and the operating system of the traditional Chinese version of Microsoft Windows XP, service pack 2, 2002.

3. Properties of Email Systems

In this study, it is assumed that all emails would be transmitted through the popular Simple Mail Transfer Protocol (SMTP) [12-14] and that users would retrieve their emails from remote server systems of the Post Office Protocol version 3 (POP3) standard [15]. In addition, most emails nowadays are of the Multipurpose Internet Mail Extensions (MIME) format [16-18] which is compatible with the SMTP standard.

However, some mail server systems do not follow the SMTP standard completely [18]. Therefore, before using an email document for data embedding, servers which do not change the content of an email body should be found, or a new SMTP server must be set up. Otherwise, data embedded in the email might be destroyed before being read and retrieved on the server at the receiver end.

According to the SMTP standard [14], the body of an email is text lines consisting of ASCII codes. The codes 0D for carriage return (CR) and 0A for line feed (LF) must appear together as 0D0A (denoted as CRLF in this paper) for use at the end of each line. A text line, if *folded*, should be limited to 78 characters, excluding CRLF. Here, *folding* means splitting a long text line into multiple shorter ones. Folding occurs when a CRLF is inserted in a line to replace a space, dividing the line into two parts.

Outlook Express, after being opened, often has a smaller window for viewing the mail content. The window width is about 70 characters. In this study, we propose to hide secret data in an email by adding ASCII control codes at the end of each text line with the resulting line being *of this width*, such that when the resulting stego-email is opened by Outlook Express, the mail body can fit the window width, thus increasing the steganographic effect. For this aim, we fold the original email lines into shorter ones, each

being 65 characters in length, leaving 5 characters at each line end as a data embedding *slot*.

Another popular protocol by which emails are accessed on a server is the Internet Message Access Protocol version 4 (IMAP4) [19]. The IMAP4 supports single web-mail servers and permits manipulations of mailboxes as remote message folders in a way that is functionally equivalent to local folders. The web mail enjoys its popularity because people can use the same client software to both surf the Internet and transmit/receive emails. And IE is probably the most popular browser for manipulating web mails. In this study, it is assumed that Outlook Express 6.0 and IE 6.0 are the preferred software for emails.

4. Embedding ASCII Control Codes into Emails

In this study, five possible ways for secret data embedding in emails have been identified using ASCII control codes. They are listed as follows.

(1) *White-space coding* --- As mentioned previously, there are many different white-space codes, each of which, when displayed, appears to be a white space, yielding the same effect as the original ASCII space code 20. For example, when the Big 5 standard uses Outlook Express, each of the three ASCII codes, 07, 09, and 0C, is displayed as a white space, as found in this study. Therefore, each of these can be used as a substitute of the space code 20 to hide data, incurring no noticeable change in the stego-email.

(2) *Inserting multiple white-space codes at text line ends* --- Multiple white-space codes can also be placed before the CRLF at the end of a text line. Since only *background* white spaces are shown after the CRLF, the extra white-space codes, though displayed as *visible* white spaces, are connected to the background white spaces, causing them to be unnoticed by the reader.

(3) *Null-space coding* --- As mentioned previously, there are many null-space codes, which are displayed as *nothing*. These can be inserted *at any position* in a line *for any repetitions* in a data hiding process without beig noticed. For example, with the UTF-8 standard using IE, the four null-space codes 1C, 1D, 1E, and 1F, are invisible.

(4) *Inserting multiple null-space codes at text line ends* --- Null-space codes can be placed repetitively at the end of a text line without being noticed since they are invisible when displayed, as in the case of (2) above.

(5) *Combining techniques of the above* --- The above techniques can be combined arbitrarily if both white-space and null-space coding are used.

From the above, it can be seen that the ASCII control codes usable for embedding secret data are *different* for different kinds of servers, browsers, and character sets. To do a systematic investigation, in this study an email file was created which included all ASCII control codes shown in Table 1 to find the SMTP server software suitable for data embedding, as well as the corresponding appearances of the ASCII control codes after being used in the different server software. The results of this investigation are given below.

First, four SMTP email servers which did not change the text contents of emails were found. These are used as *standard* SMTP servers for data embedding in this study. Their uniform resource locators (URLs) are http://cis.nctu.edu.tw, http://mis.tsint.edu.tw, http://tw.yahoo.com and http://www.hotmail.com. The first is in the Department of Computer Science at National Chiao Tung University in Taiwan, with an SMTP software of Twig 2.7.7. The department has an additional SMTP server system, Horde, for web mails. The second server is in the Department of Management Information at the Technology and Science Institute of Northern Taiwan. The SMTP software is SendMail 8.12. The third server is also in Taiwan dealing with Yahoo! Mail web mails. The last server is Hotmail, a web mail server of the Microsoft Corporation. With all four, either Outlook Express or IE is used for reading, transmitting, and receiving emails in this study. Also, the email format used is MIME 1.0, the content-type is text/plain, and the character setting is UTF-8. These formats are very commonly used and were therefore adopted in this study for the data hiding application.

After systematic testing the ASCII character set on the above-mentioned four servers, it was found that the hexadecimal ASCII control codes appropriate for data embedding under the environments of *both Outlook Express and IE* are 1C, 1D, 1E, and 1F. These four codes *all* appear to be invisible on the IE browser, and *all* are shown as white spaces in the Outlook Express window. They can so be used for data embedding according to the technique of (2) or (4) mentioned above. However, in this study it is desired to take into account *simultaneously*, instead of *respectively*, the techniques of (2) and (4) for data hiding. This results in a method of repeatedly placing these four ASCII control codes *at the ends of email text lines*. The displayed result of the stego-email yielded by this method is of no difference from that of the original cover email, achieving a steganographic effect.

More specifically, the following encoding rules are used to embed secret data into the text line ends in a cover email.

1. Encode 2-bit binary secret data “00,” “01,” “10,”

and "11" with the four ASCII codes 1C, 1D, 1E, and 1F, respectively.

2. Put the uniquely combined ASCII codes 201E before a sequence of secret data as the *start signal*, and append a copy of it at the sequence tail as the *end signal*.
3. Use the uniquely combined ASCII codes 201C to encode the 1-bit data '0,' and the uniquely combined codes 201D to encode '1.'
4. Use the uniquely combined ASCII codes 201F as a *separator* to stop the underlined display that starts from a special lexical token of the network protocol, like http, ftp, email, …, etc.

Rule 4 above is necessary because otherwise the extra white-space codes at the end of a text line, when connected to the end of a network protocol text line, will become *underlined* white spaces, like in http://cis.nctu.edu.tw, which obviously are against the purpose of steganography.

Based on the above rules, the proposed data hiding algorithm for covert communication and authentication is described in the next section.

5. Proposed Data Hiding Method for Emails

The technique proposed for embedding secret data into an email is first described as Algorithm 1 below, and then the way for transmitting the stego-email via Outlook Express or IE is described. In the following, 'an email' refers to its text body excluding the header.

Algorithm 1. ***Data embedding in an email***.
Input: a secret data file *S* and a cover email *E* long enough to hide *S*.
Output: a stego-email *E'*.
Steps:

1. Set the format of the cover email *E* to MIME 1.0, the content-type to text/plain, and the character setting to UTF-8.
2. Fold, sequentially, each long text line in *E* with more than 65 characters into a 65-character line by inserting a CRLF to replace the first space code 20 *found before the 65th character breakpoint* in the line.
3. Check every line in the resulting *E* to see if it has any special lexical token of the network protocol right before the CRLF; if so, insert a separator code 201F before the CRLF so that secret data can be inserted between the separator code and the CRLF, as described below.
4. Get a text line from *E*, starting from the first, and perform the following operations.
   4.1 Insert the start signal 201E before the CRLF which appears at the line end.
   4.2 Compute the *embedding capacity EC* between the start signal and the CRLF in the following way:
   $EC = 70 - position\ of\ CRLF\ in\ the\ text\ line$,
   which means the number of characters that can be inserted before the CRLF before the line becomes 70 characters long.
   4.3 Perform one of the following three cases (assuming that $|S|$ means the length of *S*):
   (1)if $EC \neq 0$ and $|S| > 1$, then get a pair of bits from the prefix of *S*, encode it with the corresponding code (one of 1C, 1D, 1E, 1F), insert the result before the CRLF, decrement *EC* by 1, decrement |S| by 2, and perform Step 4.3 again;
   (2)if $EC = 0$ and $|S| > 1$, then get the next text line in *E* and perform Step 4.2;
   (3)if $|S| \leq 1$, then continue.
5. Check *S* to see if there still remains a single bit *B* in *S*. If so, then:
   (1) if $EC \neq 0$, insert the code 201C before the CRLF if *B* is '0' or the code 201D if *B* is '1';
   (2) if $EC = 0$, then get a text line in *E* with nonzero embedding capacity *EC* and conduct the insertion as in Step 5(1) above.
6. Append the end signal 201E at the end of all the codes inserted in the previous steps.
7. Output the result as the desired stego-email *E'*.

After a stego-mail *E'* is obtained, it is sent to the receiver site via Outlook Express or IE like a traditional email or a web mail. To accomplish this using Outlook Express, a new email, denoted as E_n, is opened, the character set of E_n is set to UTF-8, the window size of E_n is expanded to maximum, the text body of *E' is copied* into E_n, and finally the result is sent to the receiver without encrypting it. The way using IE is similar, where IE is used to log in the selected web mail server.

6. Proposed Data Recovery Process for Emails

At the receiver end, after a stego-mail is received via Outlook Express or IE, its ASCII codes are checked for secret data extraction. The algorithm for this is described as follows:

Algorithm 2. ***Data Recovery from a stego-email text body***.
Input: a stego-email text *E'*, presumably including a secret data file *S*.
Output: the file *S*.
Steps:
1. Scan separator signals 201F in *E'* and remove all of them, if there are any.
2. Scan the resulting *E'* to find the start signal 201E in *E'* and remove it.

3. Perform the following steps.
   3.1 Get a pair of ASCII codes in order from *E*.
   3.2 If the code pair *P* is the end signal of 201E, then perform Step 4; otherwise:
      (1) if *P* is either 201C or 201D, then decode *P* to be the bit 0 or 1, respectively;
      (2) if *P* is neither 201C nor 201D, then check each code *Q* in *P* and if *Q* is one of 1C, 1D, 1E, and 1F, then decode *Q* to get the corresponding secret bit pair (one of 00, 01, 10, and 11) and remove *Q*.
   3.3 Go to Step 3.1.
4. Remove the end signal.
5. Concatenate all the decoded secret data bits extracted in the previous steps into a sequence as the desired secret data file *S* and exit.

For security consideration, the secret message may be encrypted using 3DES or AES algorithms, for example, with a secret key before the data embedding process, and the result may be embedded at random line end positions which are generated with another secret key as well as a random number generator. A reverse step can easily be performed to get the original secret message.

7. Proposed Authentication Process for Email Documents

The data embedding and extraction techniques proposed previously, in addition to being useful for the purpose of covert communication, may be used for the purpose of email authentication. More specifically, by embedding appropriately-designed codes as an *authentication signal*, the signal, when extracted, can be used to check the *fidelity* of a received email, proving that it was transmitted by a specified server and not tampered with before received. In this study, this goal is achieved by embedding an authentication signal into an email by Algorithm 1 to generate an *authenticable* stego-email. The signal is generated by using the content of an email by an SHA-1 hash function. The fidelity verification work is accomplished by matching the authentication signal extracted from a given authenticable stego-email with that computed directly from the original text content of the email. The details are described as two algorithms below.

Algorithm 3. ***Generation of an authenticable email***.
Input: a cover email *E*.
Output: an authenticable email *E′*.
Steps:

1. Fold each long text line in *E* with over 65 characters into a 65-character line by inserting a CRLF code to replace the first space code before the 65th character breakpoint.
2. Take all the ASCII code values in the modified *E* as the input to an SHA-1 hash algorithm, and take the output of the algorithm as an authentication signal *A*.
3. Use Algorithm 1 to embed *A* into the modified *E* to obtain an authenticable email as the desired output *E′*.

Algorithm 4. ***Authentication of an email***.
Input: a stego-email *E′*, presumably including an authentication signal *A′*.
Output: an authentication message about the fidelity of the displayed text content of *E′*.
Steps:

1. Take all the ASCII code values in *E′* except the special codes of 1C, 1D, 1E, 1F, 201C, 201D, 201E, and 201F, as the input of the SHA-1 hash algorithm mentioned in Algorithm 3, and take the output of the algorithm as an authentication signal *A*.
2. Extract the hidden authentication signal *A′* from *E′* by Algorithm 2.
3. Compare *A′* with *A*, and if they are identical, then output the authentication message "pass," meaning the displayed text content of *E′* is genuine; else, the message "fail," meaning the reverse.

8. Experimental Results

Figures 1 through 4 illustrate some experimental results of applying Algorithms 1 and 2 for covert communication using Outlook Express. Figure 1 shows part of the content of a 9.3KB cover email. Figure 2 shows part of the content of the stego-email (12.7KB) obtained by applying Algorithm 1 with the cover email as the input. This content was displayed with Outlook Express by a receiver with email address tmp168@mis.tsint.edu.tw, to whom the stego-email was sent. From Figure 2, it is seen that there was no difference in the stego-email when compared with the cover email. Figure 3 shows the content of the 1.07KB secret data file embedded in the stego-email. And Figure 4 shows the content of the 1.07KB secret data file extracted from the stego-email shown in Figure 2 by applying Algorithm 2. The two file contents can be seen to be the same. These results show that the proposed method of data hiding and recovery is feasible.

Figures 5 through 9 illustrate some additional experimental results of applying the proposed algorithms using IE. All password portions in the emails in these figures were blackened for protecting the privacy of the mail owners. Figure 5 shows the content of a 2.42KB cover email. Figure 6 shows the content of the corresponding stego-email (2.54KB) generated by Algorithm 1. Figure 7 shows part of the

content of the stego-email seen as a web mail in IE at a receiver site with address gis87809@cis.nctu.edu.tw. Figure 8 shows the content of the original secret data file with 27 bytes. Figure 9 shows the content of the secret data file that was extracted from the stego-email shown in Figure 7. Again, the original and the extracted secret data can be seen to be identical.

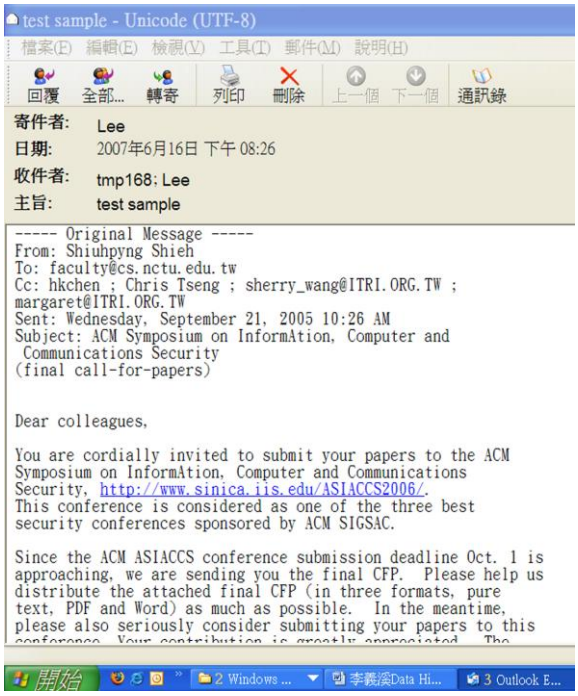


Figure 1: Part content of a cover email

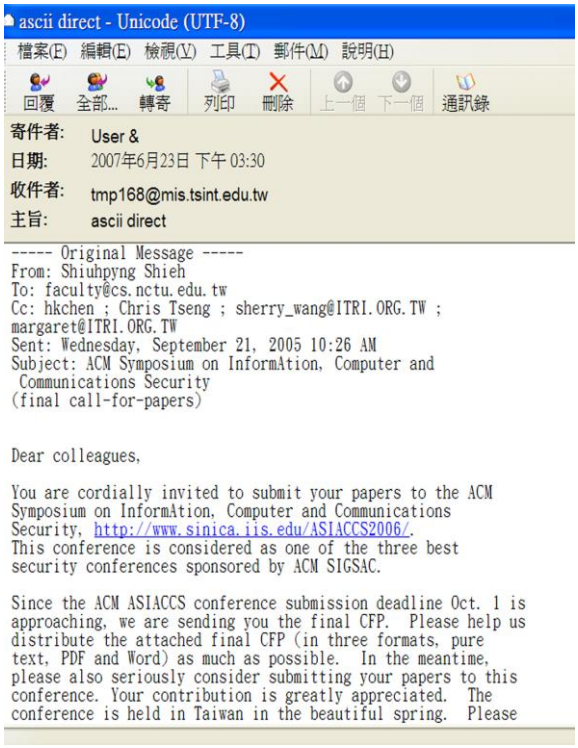


Figure 2: Part content of the stego-email generated from Figure 1

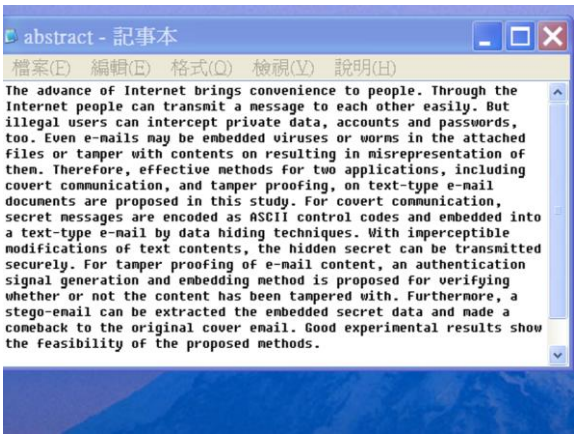


Figure 3: Content of an embedded secret data file

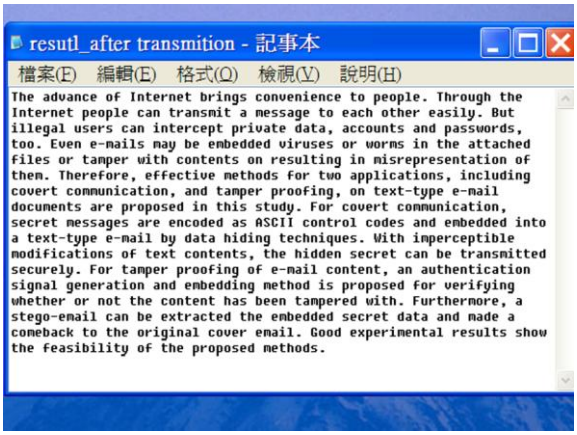


Figure 4: Content of the extracted secret data file

Both the transmitter's and the receiver's operations were performed on the same server in the experiments above. Experiments were also conducted on different servers for the transmitter's and the receiver's operations. For example, one server was the mail server at Yahoo! in Taiwan, and the other a mail server in the Department of Computer Science at National Chiao Tung University in Taiwan. The results remained unchanged.

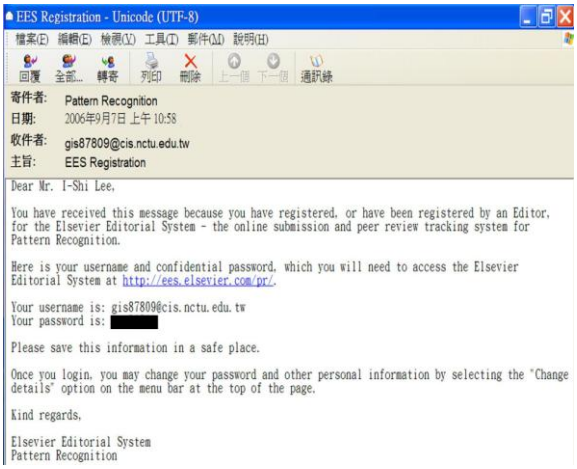


Figure 5: Content of a cover email

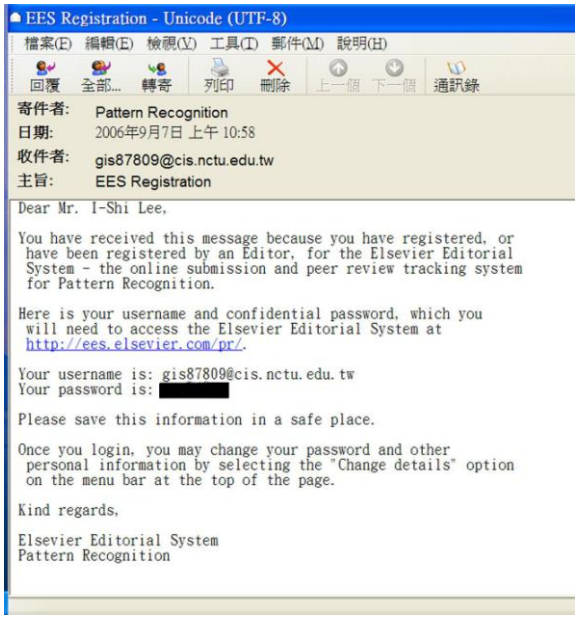


Figure 6: Content of the stego-email generated from Figure 5 before being transmitted

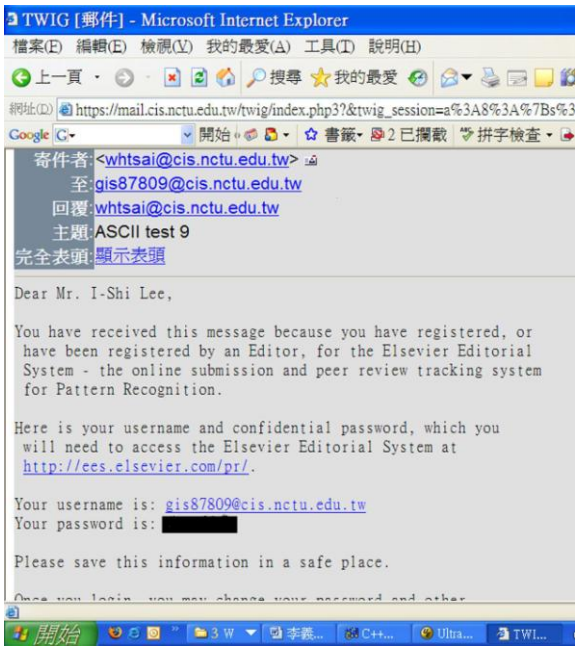


Figure 7: Part content of the stego-email received and displayed in IE

Figure 8: Content of the original secret file

Figure 9: Content of the extracted secret file

Figures 10 to 13 illustrate some experimental results of applying the proposed email authentication method. Figure 10 shows the content of a stego-email which was generated by Algorithm 3. The password portion in the stego-email was also blackened for protecting privacy. The embedded secret data are invisible to a casual reader. Figure 11 shows part of the content of the stego-email file after being received via Outlook Express, and the authentication result of "pass". Figure 12 shows part of the content of the stego-email after being received by IE, and the authentication result of "pass," too. Figure 13 shows part of the content of the stego-email file after being received by IE, and the authentication result of "fail," since the content has been tampered with (the word "Lee" has been changed to "lee"). These results show that the proposed email authentication method is effective.

9. Conclusions and Discussions

In this paper, a method is proposed to embed secret data into emails using the ASCII codes of the traditional Chinese version of Microsoft Windows XP, service pack 2, 2002. After systematic testing all the ASCII codes on various email server software systems and standards, four special ASCII control codes 1C, 1D, 1E, and 1F were found to be invisible at the line ends of email texts on the SMTP email server in the environment of Outlook Express or IE. A technique has been proposed to utilize these special codes to encode secret data, which is a combination of five coding rules found in this study. Each stego-email can be transmitted to a receiver, and read as a normal email. Extra long lines are folded to be of a proper length for normal displays on email servers to increase steganographic effects. The experimental results proved the feasibility of the proposed method.

In this study, 2-bit secret data were embedded into a white space of a text email. Comparing to other methods proposed by Bender et al. [2] and Chang and Tsai [11] in which on average each secret bit needs 1.5 white spaces to encode (one white space representing a "0," and two white spaces representing a "1," leading to the average of $1\times0.5+2\times0.5 = 1.5$ spaces for a secret bit), the proposed method needs only 0.5 white space for each secret bit (one ASCII code representing 2 secret bits), which is threefold increase of the embedding capacity.

The proposed data hiding method may be used on the four servers mentioned previously. However, not all mail servers fully follow the SMTP standard. Instead, some mail servers have their own ways of management, like Gmail and Yahoo! Mail. They delete redundant spaces and undefined characters. So, the proposed method is inapplicable to these two servers. Other applicable techniques should be

investigated, and are left for further study. Another topic worth future investigation is applying the proposed data hiding technique to check the integrity of an email, in addition to the fidelity check scheme proposed in this study. Finally both the covert communication and authentication works of this study may be extended to dealwith web pages.

Acknowledgement

This work was supported partially by the NSC Advanced Technologies and Applications for Next Generation Information Networks (II) – Sub-project 5: Network Security, Project No. NSC-94-2752-E-007-001-PAE. And partially by the NSC project NSC94-2422-H-468-001.

References

[1] D. C. Lou and J. L. Liu, 2002, "Steganographic Method for Secure Communications," *Computers and Security*, Vol. 21, No. 5, pp. 449-460.

[2] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, 1996, "Techniques for data hiding," *IBM System Journal*, Vol. 35, No. 3 & 4.

[3] S. Katzenbeisser and F. A. P. Petitolas, 2000, *Information Hiding Techniques for Steganography and Digital Watermarking*, Artech House, Boston, MA.

[4] Low, S. H., N. F. Maxemchuk, and A. M. Lapone, 1998, "Document Identification for Copyright Protection Using Centroid Detection," *IEEE Transactions on Communication*, Vol. 46, No. 3, pp. 372-383.

[5] J., Brassil, N. F. Maxemchuk, and L. O'Gorman, 1994, "Electronic Marking and Identification Techniques to Discourage Document Copying," *Proceedings of 13th Annual IEEE Conference on Computer Communications(INFOCOM'94)*, pp.1278-1287.

[6] P. Wayner, 1995, "Strong Theoretical Steganography," *Cryptologia*, Vol. XIX/3, pp. 285-299.

[7] G. Cantrell and D. D. Dampier, 2004, "Experiments in Hiding Data Inside the File Structure of Common Office Documents: A Stegonography Application," *Proceedings of 2004 International Symposium on Information and Communication Technologies*, Las Vegas, Nevada, USA, pp. 146-151.

[8] N. F. Johnson, Z. Duric, and S. Jajodia, 2001, *Information Hiding: Steganography and Watermarking-Attacks and Countermeasures*, Kluwer Academic Publishers, Boston, MA, USA.

[9] R., Anderson, R. Needham, and A. Shamir, 1998, "The Steganographic file System," *Information Hiding: Second International Workshop*, Portland, Oregon, USA.

[10] T. G., Handel, and M. T. Stanford, 1996, "Hiding Data in the OSI Network Model," *Proceedings of First International Workshop on Information Hiding*, Cambridge, UK, pp. 23-38.

[11] Y. H. Chang and W. H. Tsai, 2003, "A steganographic method for copyright protection of HTML documents," *Proceedings of 2003 National Computer Symposium*, Taichung, Taiwan.

[12] J. B. Postel, 1982, "Simple Mail Transfer Protocol," *STD 10, RFC 821, IETF*.

[13] D. Crocker, 1982, "Standard for the Format of ARPA Internet Text Messages," *STD 11, RFC 822, IETF*.

[14] P. Resnick, 2001, "Internet Message Format," *RFC 2822, IETF*.

[15] J. G. Myers and M. T. Rose, 1996, "Standard Post Office Protocol - Version 3," *STD 53, RFC 1939, IETF*.

[16] N. Freed and N. Borenstein, 1996, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies," *RFC 2045, IETF*.

[17] N. Freed and N. Borenstein, 1996, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types," *RFC 2046, IETF*.

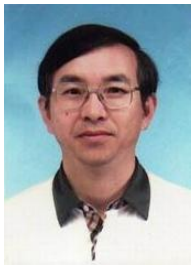
[18] N, Freed. and N. Borenstein, 1996, "Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples," *RFC 2049, IETF*.

[19] M. Crispin, 2003, " Internet Message Access Protocol - Version 4rev1," *RFC 3501, IETF*.

Biographies

I-Shi Lee was born in Taipei, Taiwan, R.O.C., in 1961. He received the B. S. degree in electronic engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, Republic of China in 1987, the M. S. degree in the Department of Computer Science and Information Science at National Chiao Tung University in 1989, and the Ph.D. degree in the Institute of Computer Science and Engineering, College of Computer Science from National Chiao Tung University in 2008.

In 1992, he joined the Department of Management Information at Northern Taiwan Institute of Science and Technology where he has been a lecture till now. His recent research interests include pattern recognition, watermarking, and image hiding.



Professor Wen-Hsiang Tsai was born in Tainan, Taiwan on May 10, 1951. He received the B. S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan in 1973, the M. S. degree in electrical engineering (majoring in computer science) from Brown University, Providence, Rhode Island, U. S. A. in 1977, and the Ph. D. degree in electrical engineering (majoring in computer engineering) from Purdue university, West Lafayette, Indiana, U. S. A. in 1979.

Dr. Tsai joined the faculty of National Chiao Tung University (NCTU), Hsinchu, Taiwan in November 1979. He is currently an NCTU Chair Professor in the Department of Computer Science. Professor Tsai has been an Associate Professor of the Department of Computer Engineering (now Department of Computer Science) and the Acting Director of the Institute of Computer Engineering. In 1984, he joined the Department of Computer and Information Science (now also Department of Computer Science), and acted as the Department Head from 1984 through 1988. He was also the Associate Director of the Microelectronics and Information System Research Center from 1984 through 1987, the Dean of General Affairs from 1995 to 1996, the Dean of Academic Affairs from 1999 to 2001, the Acting Dean of the College of Humanities and Social Science in 1999, and the Vice President of the University from 2001 to 2004. From August 2004 to July 2007, he was the President of Asia University, Taichung, Taiwan.

Outside the campus, Professor Tsai has served as a Consultant to many major research institutions in Taiwan, including the Chun-Shan Institute of Science and Technology, the Industrial Technology Research Institute, and the Information Industry Institute. He has acted as the Coordinator of Computer Science in the Engineering Division of the National Science Council, and a member of the Counselor Committee of the Institute of Information Science of Academia Sinica in Taipei. He has been the Editor of several academic journals, including Computer Quarterly (now Journal of Computers), the Proceedings of National Science Council, the Journal of the Chinese Engineers, the International Journal of Pattern Recognition and Artificial Intelligence, the Journal of Information Science and Engineering, and Pattern Recognition. He was the Editor-in-Chief of the Journal of Information Science and Engineering from 1998 through 2000. From 2006, he is the Editor-in-Chief of the Asian Journal of Health and Informaiton Science.

Professor Tsai's major research interests include image processing, computer vision, virtual reality, and information copyright and security protection. So far he has published 339 academic papers, including 129 journal papers and 210 conference papers. He is also granted eight R. O. C. and U. S. A. patents. Dr. Tsai has supervised the thesis studies of 31 Ph. D. students and 143 master students. Dr. Tsai is a senior member of the IEEE, a member of the Chinese Image Processing and Pattern Recognition Society, and the International Chinese Computer Society. He served as the Chairman of the Chinese Image Processing and Pattern Recognition Society of Taiwan from 1999 to 2000. He was the Chair of the Computer Society of IEEE Taipei Section from 2003 to 2008.

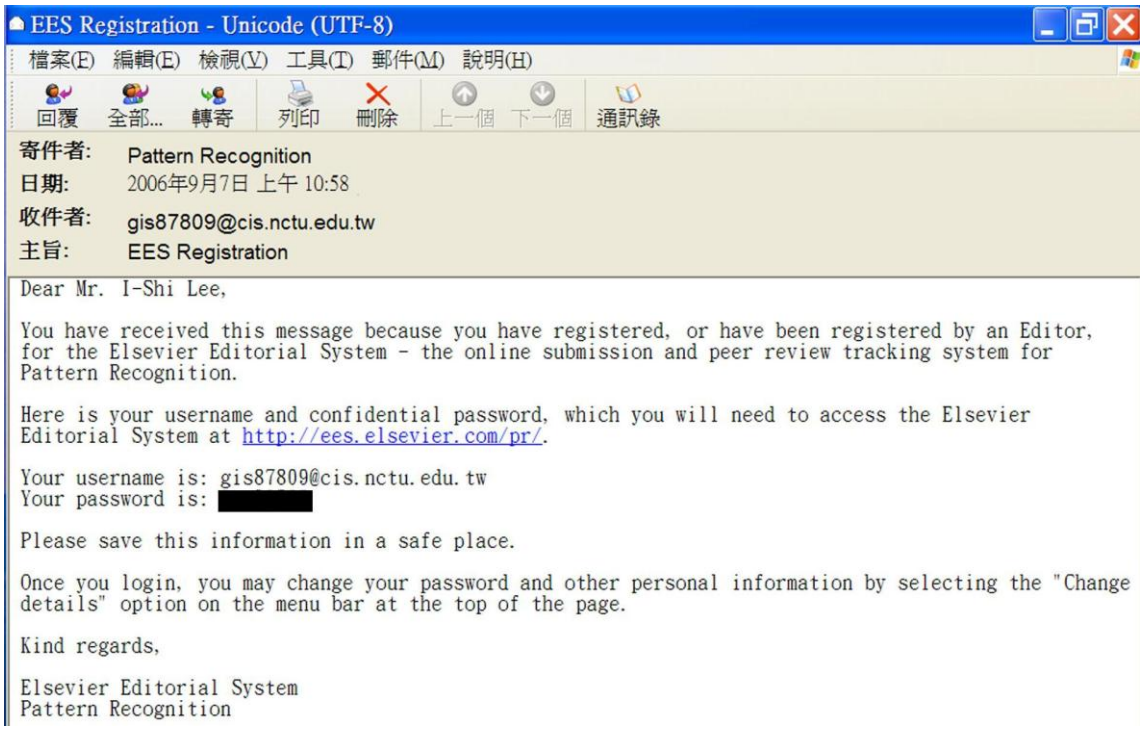


Figure 10: Content of a stego-email for authentication before transmission

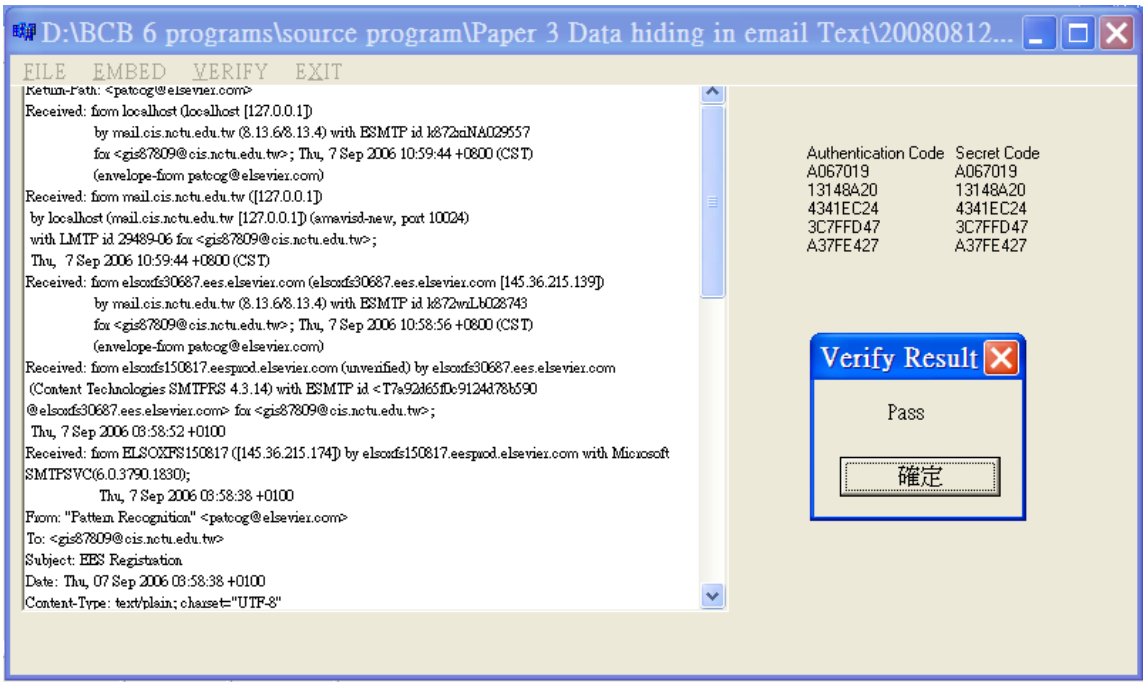


Figure 11: Authentication result of "pass" after receiving a stego-email by Outlook Express

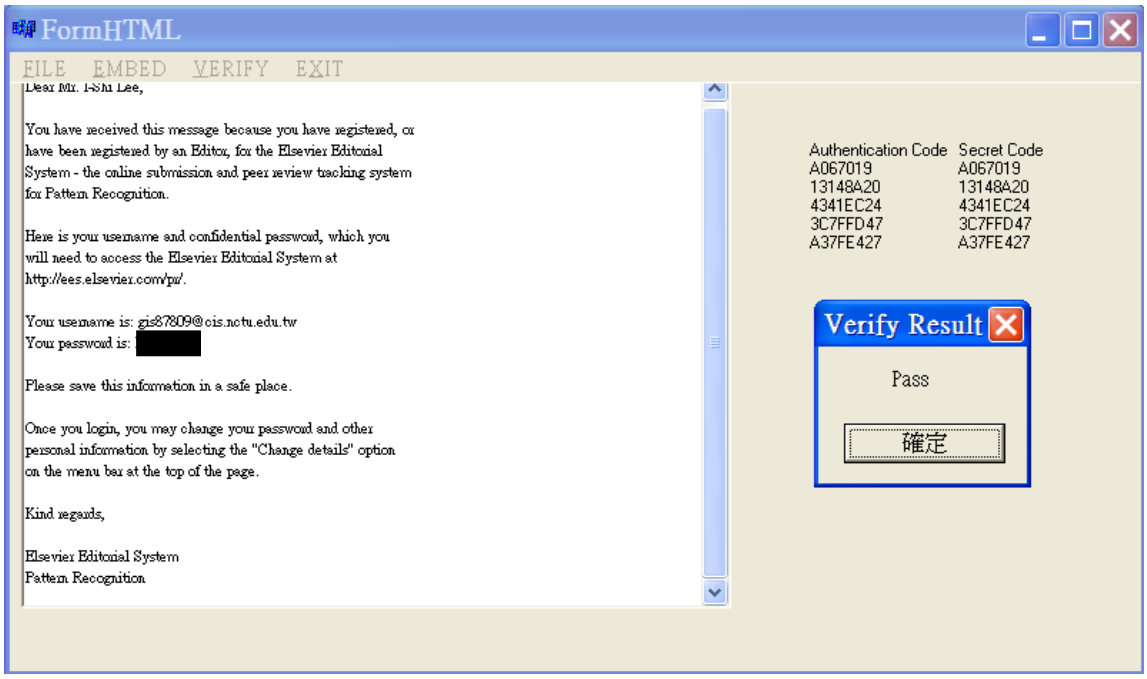


Figure 12: Authentication result of "pass" after receiving a stego-email by IE

Figure 13: Authentication result of "fail" after receiving the stego-email by IE. The word "Lee" in the content has been modified to be "lee."