A novel covert channel based on the IP header record route option

Zouheir Trabelsi* and Hesham El-Sayed

College of Information Technology,
United Arab Emirates University,
P.O. Box 17555, Al-Ain, UAE
E-mail: trabelsi@uaeu.ac.ae E-mail: helsayed@uaeu.ac.ae
*Corresponding author

Lilia Frikha

College of Telecommunications (SupCom),
The University of Carthage,
Cité Technologique des Communications,
Route de Raoued Km 3,5 – 2083 El Ghazala, Ariana, Tunisia
E-mail: lilia_frikha@yahoo.fr

Tamer Rabie

College of Information Technology,
United Arab Emirates University,
P.O. Box 17555, Al-Ain, UAE
E-mail: tamer@uaeu.ac.ae

Abstract: In this paper we propose a novel covert channel for exchanging secret information, based on the IP header record route options. Instead of encrypting a secret message or embedding it into a multimedia object, as in traditional steganography, we process the entire message and generate several IP packets with different types to carry the secret information. Thereby we foil an eavesdropper who is primarily applying statistical tests to detect encrypted channels. We show that our approach provides more protection against steganalysis and sniffing attacks, and gives a covert channel capacity which is an order of magnitude higher than traditional methods.

Keywords: covert channel; hidden information; IP header record route option; steganalysis; traceroute.


Biographical notes: Z. Trabelsi is an Assistant Professor at the College of Information Technology, United Arab Emirates University, UAE. He received his PhD in Computer Science from the Tokyo University of Technology and Agriculture, Japan. His current research interests include networking security, intrusion detection and prevention, TCP/IP covert channels, and network protocol design.

Copyright © 2007 Inderscience Enterprises Ltd.
H. El-Sayed is an Assistant Professor at the College of Information Technology, United Arab Emirates University, UAE. He received his PhD in Computer Engineering from the Carleton University, Canada. His current research focuses on network security, protocol design, integration of wireless networks, and sensor networks.

L. Frikha is currently a PhD student at the High College of Telecommunications of Tunisia (SupCom). Her research areas are mainly networking security, TCP/IP covert channels, network protocol design, and intrusion detection and prevention.

T. Rabie is an Associate Professor at the College of Information Technology, United Arab Emirates University, UAE. He received his PhD in Computer Science from the University of Toronto, Canada. His research interests include digital image processing, computer vision, virtual reality, and their application to the field of intelligent transportation systems.

1 Introduction

Rapid developments in cryptography, cryptographic standards, and relaxed export control on encryption schemes have made cryptographic software widely available, user friendly and, in some cases transparent, to end-users. The relative ease with which secure channels of communication can be created between two parties using cryptography, coupled with the explosion in network traffic, has posed a problem to the surveillance operations typically carried out by law enforcement and intelligence agencies. In order to keep up with the growth in communication traffic and secure channels, massive eavesdropping campaigns, such as Carnivore and Echelon, have been developed. Carnivore, for example, is a combination of hardware and software, which collects all electronic packets that are sent to and from the Internet Service Provider where it is installed. Following this ‘collection’ procedure, several filters are applied to isolate e-mails with certain keywords which could be further analysed offline. These filters could potentially include the performance of statistical tests to isolate encrypted e-mails for offline cryptanalysis (Radhakrishnan et al., 2002).

Pervasive eavesdropping like that performed by Carnivore leads to the need for covert channels or Steganography, in which communicating entities hide their communication channel itself from eavesdroppers. Steganography is the technique of hidden communication. It relies on hiding covert message in unsuspected multimedia data. It is generally used in secret communication between acknowledged parties. It is a method of encryption that hides data among the bits of a covert file, such as a graphic or an audio file. The technique replaces unused or insignificant bits with the secret data. A covert channel is a mechanism that can be used to communicate data across network, or between processes within the system, in a manner that goes unnoticed (US DOD, 1985). An effective covert channel is the one that is undetectable by the adversary and can provide a high degree of privacy. The goal of the covert channel is to communicate data from one host to another host in such a way that the receiving host can detect the data but the eavesdropper will not even get a hint that some secret data is being communicated. Some features of the TCP/IP protocol suite can be used to send covert messages as discussed in Rowland (1997). Encrypted or non-encrypted information can be
encapsulated within normal TCP/IP packets. The TCP/IP header information can also be modified to encode secret messages. There are some fields in the packet headers that are not used by the current communication networks (TBD address REV_B comment 3). These fields can be used as message carriers.

These covert channels are an immense cause of security concern because they can be used to pass malicious messages. These messages could be in the form of computer viruses, spy programs, terrorist messages, etc. Therefore, detecting these covert channels is an important issue that needs to be addressed (Fisk et al., 2002). However, covert channels can also be used to exchange hidden information, such as e-commerce transaction data or governmental confidential information, so that a hacker or any one spying the communication channel will not be able to detect that the captured packets carry hidden information. The sheer volume of internet traffic provides a higher bandwidth vehicle for covert communications which leads to a plethora of applications.

This paper relates the areas of Steganography, network protocols and security for practical data hiding in communication networks employing TCP/IP. It proposes a novel covert channel, based on the IP header record route options, for sending hidden information. It will be demonstrated that the hidden information exchanged over the covert channel is protected against steganalysis and sniffing. That is, a hacker or any one spying the communication will not notice the existence of hidden information in the packets exchanged.

The rest of the paper is organised as follows. Section 2 provides the necessary background information to understand the principle of the proposed covert channel. Section 3 discusses the related work in the literature. Section 4 discusses the proposed cover channel and presents a mechanism to protect the channel against steganalysis and sniffing. It also presents an example of how a hidden message is inserted in the covert memory and shows a comparison between the covert memories available in several existing covert channels. Section 5 presents a friendly tool, implemented to demonstrate the proposed covert channel. The tool provides the user with a friendly interface to send and receive hidden messages. Finally, Section 6 provides the conclusion and future work.

2 Background

In order to introduce the terms used in this paper and lay the groundwork for what follows, we will explore the different fields and options in the IP protocol header. Once a TCP, UDP or ICMP packet is generated, it must be sent across the network. The packet will be passed to the Network layer for end-to-end packet delivery. The IP protocol is the most commonly used protocol in the network layer today, and is used for almost all traffic moving across the internet. Upon receiving a TCP, UDP or ICMP packet, the IP protocol generates an IP header which includes the fields shown in Figure 1. Then, the IP header is added to the front of the TCP, UDP or ICMP packet to create the resulting IP packet, which will be used to carry the entire contents (IP header, TCP/UDP/ICMP header, and application-level data) across the network.
A novel covert channel based on the IP header record route option

Figure 1  The IP header structure

The 4-bits HLEN field is the IP header length. An IP header without options has a length of 20 bytes. However, an IP header with options may have a length of up to 60 bytes. It is important to indicate that most of the IP packets in the internet do not use the Option field. Therefore, in most cases, a covert channel can use the unused 40 bytes related to the Option field in an IP header to insert hidden information in IP packets.

2.1 The fields of the IP header option

The IP Option field in the IP header is not required in every IP datagram. Options are included primarily for network testing or debugging. Options processing is an integral part of the IP protocol, and all standard implementations must include it.

Figure 2 shows the structure of the IP header option. The field Code indicates the type of the option in the IP header. The field Length indicates the size of the field Option. The Pointer field plays a particular function, depending on the type of the option. There are eight possible types of options in an IP datagram. The four most used options are:

- **loose source routing**: used to route a datagram along a specific path
- **record route**: used to trace a route
- **strict source routing**: used to route a datagram along a specified path
- **internet timestamp**: used to record timestamps along a route.

Figure 2  The IP header option structure

For the proposed covert channel in this paper, we are only interested in the record route option.
2.2 The record route option in an IP header

The record route option in an IP datagram allows the source host to create an empty list of IP addresses and arrange for each router that handles the datagram to add its IP address to the list. Figure 3 shows the format of the record route option. The Code field is set to the value 7. The Length field specifies the total length of the record route option in an IP datagram, including the first three bytes. The Pointer field specifies the offset within the record route option of the next available slot. That is, it specifies the position in the record route option where the next gateway can insert its IP address.

Figure 3 The format of the record route option in an IP datagram

Whenever a router handles an IP datagram that has the record route option set, the router adds its IP address to the record route list. It is clear that enough space must be allocated in the record route option by the original source host to hold all the IP addresses of the routers. To add its IP address to the list, a router first compares the values in the Pointer and Length fields. If the value in the Pointer field is greater than the value in the Length field, this means that the list is full, the router then forwards the IP datagram without inserting its IP address. If the list is not full, the router inserts its 4-bytes IP address at the position specified by the Pointer field, then increments the Pointer by four. When the IP datagram reaches its destination, the destination host extracts and processes the record route list of IP addresses.

2.3 Classes of IP addresses

In order to provide the flexibility required to support different size networks, the internet designers decided that the IP address space should be divided into three different address classes – Class A, Class B and Class C (Mogul and Postel, 1985; Gerich, 1993). This is often referred to as 'classful' addressing because the address space is split into three predefined classes, groupings, or categories. Each class fixes the boundary between the network-prefix and the host-number at a different point within the 32-bit address. The formats of the fundamental address classes are illustrated in Figure 4.

In addition to the three most popular classes, there are two additional classes. Class D addresses have their leading four-bits set to 1-1-1-0 and are used to support IP Multicasting. Class E addresses have their leading four-bits set to 1-1-1-1 and are reserved for experimental use.
3 Related work

The concept of a covert channel was first introduced by Lampson (1973) as a channel that is used for information transmission. Then Girling (1987) analyses covert channels in a network environment. His work focuses on Local Area Networks (LANs) in which three obvious covert channels (two storage channels and one timing channel) are identified. The first uses the bits reserved for addresses, the second uses the bits reserved for the length and the third uses the time difference between the packets. Wolf (1989) presents results applied to LAN protocols. He highlights the relationship between covert storage channels and protocol format, and the link between covert timing channels and protocol procedure elements taking into account the frame layouts of the LAN protocols. Covert storage channels utilise the reserved fields, padding fields and undefined fields of the frames. Handel and Sandford (1996) take a broader perspective and focus on covert channels within the general design of network communication protocols. They employ the Open System Interconnection (OSI) network model as a basis for their development in which they characterise system elements having the potential to be used for data hiding.

Covert channels are discussed more generally in a variety of papers. A generalised survey of information-hiding techniques is described in ‘Information Hiding – A Survey’, (Kuhn et al., 1999). Theoretical issues in information hiding are considered in Cachin (1998) and Anderson and Petitcolas (1998). McHugh (1995) provides a wealth of information on analysing a system for covert channels.

Many covert channels have been identified in the IP and TCP protocols (Murdoch and Lewis, 2005), using fields like: the IP identification field, the TCP initial sequence number field, the TCP acknowledge sequence number field, windowing bits, and protocol identification field (Kamran, 2002; Kamran and Kundur, 2002; Rowland, 1997). These papers focused on finding places where covert data could potentially be sent but did not work out the details of how to send it. Three examples of these covert channels are briefly discussed here:
Covert channels using the Initial Sequence Number field. The Initial Sequence Number field (ISN) of the TCP/IP protocol suite enables a client to establish a reliable protocol negotiation with a remote server. As part of the negotiation process for TCP/IP, several steps are taken in what is commonly called a ‘three way handshake’. The sequence number field serves as a perfect medium for transmitting clandestine data because of its size (a 32 bit number). There are a number of possible methods. The simplest is to generate the sequence number from the ASCII character that will be encoded.

Covert channels using the TCP Acknowledge Sequence Number field. This method relies upon basic spoofing of IP addresses to enable a sending machine to ‘bounce’ a packet of information off of a remote site and have that site return the packet to the real destination address. This has the benefit of concealing the sender of the packet as it appears to come from the ‘bounce’ host. This method could be used to set up an anonymous one-way communication network that would be difficult to detect, especially if the bounce server is very busy. This method relies on the characteristic of TCP/IP where the destination server responds to an initial connect request (SYN packet) with a SYN/ACK packet containing the original initial sequence plus one (SYN + 1).

Covert channels based on the packet sorting. This method presented by Kamran (2002) deals with the use of packet ordering to convey covert information. The possible ways to arrange objects in a set are surprisingly complex and offer a correspondingly large opportunity for Steganography. Changing the order of the packets requires no change in the packet content (i.e., the payload and the headers are not affected). Therefore no major modifications are expected either in the protocol definition/design or in the overall system in order to implement a data-hiding scenario. The sorting/resorting process holds a surprisingly large amount of information. Based on these facts, data hiding was achieved based on packet sorting and resorting processes at source and destination, respectively. Kamran (2002) and Kamran and Kundur (2002) also discussed a covert channel based on the flags bits (URG, ACK, PSH, RST, SYN, FIN) in the TCP header. This covert channel offers only few bits as a covert memory per TCP packet. Techniques for detecting covert channels, as well as possible places to hide data in the TCP stream, are discussed in UC Davis (1999). In Abad (2001), the idea of using IP checksums for covert communication is discussed. Katzenbeisser and Petitcolas (2000) have also observed the potential for data hiding in the TCP/IP protocol suite. They use the term Internet Steganography for this potential scenario and indicate that the ongoing research work includes the embedding, recovering and detecting information in TCP/IP packet headers. In Giffin et al. (2002), the idea of hiding data in TCP timestamps is discussed. By imposing slight delays on the processing of selected TCP packets, the low order bits of their timestamps can be modified. The low bit of the TCP timestamp, when modified in this way, provides a covert channel.

Also, some research works propose to use ICMP packet (Postel, 1984) to carry hidden information in the ICMP header (Singh et al., 2003; Kamran, 2002). These covert channels involve putting hidden data and messages in the data fields of the ICMP packets, mainly in Ping ICMP packets. Obviously, the existence of hidden data and
messages in such data fields can be easily identified. However, such covert channels exploit the fact that network devices usually do not apply filters on the data fields of the ICMP headers. Three examples of ICMP based covert channels are briefly discussed here:

- **ICMP echo request and ICMP echo reply messages.** The optional data field allows having variable length data to be returned to the sender. IP options like router alert, record route and time stamp can be used to encapsulate ICMP echo request message. This provides a possibility to have covert channel between the communicating parties. Moreover, network devices usually do not filter the contents of ICMP echo traffic if ICMP echo traffic is allowed.

- **ICMP address mask request.** An ICMP address mask request is sent from a host to a specific router on the LAN or broadcasted to all routers of the LAN. The request is filled with zeros in the 32-bit address mask field. This can be used to have covert communication from host to router(s) on the same LAN.

- **Router solicitation.** After booting, a host sends a solicitation request to a router on the LAN network. Immediately, the router responds with an ICMP message router advertisement packet. It has a 32 bit reserved word. These reserved bits can be made to use for covert communication for a specific scenario.

Unfortunately, most of the existing covert channels in literature are not efficient and/or practical. They either provide very limited covert memory such as TCP-based covert channels, or are not robust enough against steganalysis such as ICMP-based covert channels. The covert channel proposed in this paper addresses the above drawbacks and provides a more practical and robust covert channel for hiding information.

4 A covert channel based on the record route option

The idea behind the proposed covert channel is to allow a source host to use the available bytes in the record route option to insert hidden information, and at the same time prevent any router (along the path to the destination host) from inserting its IP address to keep the hidden information intact. Figure 5 shows the structure of an IP packet without, and with, the record route option in an Ethernet network. When the IP header option designates a record route, the Code and Pointer fields should be set to the standard values 7 and 4, respectively. Also, the maximum value in the Length field should be 39 (Table 1). On its way to its destination, any packet with such an IP header option would ask each router to write its IP address in the 4-bytes field pointed by the Pointer field (Figure 6). Then, the value of the Pointer field in the IP header option is increased by 4. So, the next router would write its IP address in the next 4-bytes field in the IP header option. However, if the value of the Pointer field becomes greater than the value of the Length field, then no more routers can write their IP addresses. Therefore, we may establish a covert channel for sending hidden messages if the initial value of the Pointer field is greater than the value of the Length field (Figure 7(a)), or just greater than the length of the hidden message (Figure 7(b)).
Figure 5  Structure of an example IP packet without and with the record route option

Figure 6  A normal record route option header

Table 1  The standards values of the fields of the IP option

<table>
<thead>
<tr>
<th>Fields</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code</td>
<td>7</td>
</tr>
<tr>
<td>Maximum length</td>
<td>39</td>
</tr>
<tr>
<td>Pointer</td>
<td>4</td>
</tr>
</tbody>
</table>

Figure 7  The different values of the Pointer field used for the covert channel

(a)
A novel covert channel based on the IP header record route option

Figure 7 The different values of the Pointer field used for the covert channel (continued)

If we set the initial value of the Pointer field greater than the value of the Length field, then no router can write its IP address. In this case, we can use all the 36 bytes of the IP header option to put hidden data or messages. However, if we set the initial value of the Pointer field just greater than the length of the hidden message, then a number of routers can write their IP addresses in the remaining bytes of the IP header option.

4.1 Examples of ICMP packets with record route options

The packet generator of the Sniffer CommView is used to generate an ICMP Ping packet with a record route option. However, we set the value of the Pointer field greater than the value of the Length field. Figure 8 shows that the IP addresses of the source and destination hosts are 172.16.16.3 and 172.16.16.20, respectively. In addition, the hidden message written in the Option field is: “This is a covert channel”, whose length is 24 bytes. Therefore, the value of the Length field is 39 bytes. The value of the Pointer field is set to 28, in order to force any router to write its IP address in the 4-bytes field that just follows the hidden message. Using the Sniffer Ethereal, the contents of the two IP options in the packet sent to the destination host and in the packet received from the source host are decoded. Figure 8 illustrates this, and shows that the first router (which is the destination host in our case) writes its IP address just after the hidden message. This demonstrates that the manipulation of the value of the fields in the record route option in an IP header, allows us to propose a covert channel for sending hidden information. This covert channel has the following features:

- **Considerable covert memory.** In the proposed covert channel, we have nearly 40 bytes of covert memory per packet. This provides more flexibility compared to the existing TCP-based covert channels such as in Rowland (1997), which offer a maximum of 4 bytes of covert memory per packet.
- **Flexibility.** The proposed covert channel may only use ICMP or UDP packets to exchange hidden information. This removes any restrictions imposed by the use of TCP packets (such as synchronisation, flow and congestion control) as in the case of TCP-based covert channels (Rowland, 1997).
• **Undetectability.** Inserting the hidden information in the record route options will in general not alert users who are analysing the traffic about the presence of the hidden information in those packets. Most of these users would assume that this hidden information represents a valid list of IP addresses of the routers along the connection path. However, an advanced steganalysis process may be able to notice that these IP addresses are not valid and generate a suspicious situation. In Section 4.2, we propose a mechanism to further protect the covert channel from such steganalysis. But, the available covert memory will be reduced in order to offer protected covert channel against steganalysis.

**Figure 8** ICMP packets with record route options captured by **Ethereal sniffer**

4.2 **Protection against steganalysis and sniffing**

The information in the covert channel is packaged in the form of IP addresses of routers. However, it is possible for one to verify the validity of these IP addresses in the connection path which immediately offers a means for steganalysis. For example, if we want to send the hidden message ‘RDV at 9pm’ in an IP packet, then the contents of the record route option would appear as shown in Figure 9. However, a steganalysis process may identify that the IP addresses in the record route option (82.68.86.32, 97.116.32.57, 112.109.0.0) are not valid IP addresses.

**Figure 9** The contents of the record route option
To protect the scheme from such a potential steganalysis process, it is clear that the IP addresses used in the record route option should look-like valid router IP addresses. Hence, a mechanism is proposed to generate packets carrying hidden information and satisfying the above condition.

The proposed mechanism is based on two steps. The purpose of the first step is to collect the IP addresses of the routers that will most likely be in the connection path. The purpose of the second step is to compute the number of IP packets needed to carry the hidden information and generate the contents of the record route options in these packets.

**Step 1: Collection of the IP addresses**

Before generating any packet carrying hidden information, we collect the list of IP addresses of the routers that will most likely be in the connection path. Unix command ‘traceroute’, Windows command ‘tracert’ (Malkin, 1993), or any program that provides the same functionality can be used to collect such a list as shown in Figure 10. In theory, this path may not be identical for all packets sent to the same destination. However, in practice all packets that belong to the same flow always follow very similar paths, if not the same path. So, this should not raise any concern.

**Figure 10** Unix command ‘traceroute’

![Figure 10](image)

**Step 2: Generation of the contents of the record route options**

To generate packets carrying hidden information and protect them from steganalysis process, the following two requirements should be satisfied: first, the IP addresses inserted in the record route options should look-like valid router IP addresses. That is, they should be very similar to the router IP addresses collected by commands such as ‘traceroute’ or ‘tracert’. Second, the hidden information should be included in the IP addresses that are inserted in the record route options.

Hence, we developed an algorithm for generating IP packets carrying the hidden information and satisfying the above two requirements. The algorithm takes two parameters as inputs:

- a hidden message of $k$ characters; $HM = \{c_1, \ldots, c_k\}$
- the collected list of IP addresses of the routers that most likely will be in the connection path. This list can have a mixture of Class A, B and C IP addresses: $List_{IP} = (\{IP_1, \ldots, IP_{nB}\}, \{IP_1, \ldots, IP_{nC}\})$, where $n_B$ = number of Class A and B IP addresses, and $n_C$ = number of Class C IP addresses. It is important to note that a maximum of nine IP addresses can be inserted in a record route option, since the maximum available space in the record route option is 39 bytes, and each IP address needs four bytes.
As output, the algorithm produces:

- The number of IP packets that are needed to carry the hidden text and the contents of
  the record route option in each packet to be sent.

The algorithm is defined in Figure 11.

**Figure 11** Algorithm generatePackets

```
Algorithm generatePackets (HM, List_IP)
Step 1: calculate the number of IP packets needed to hide HM using the following formula:

\[ P(k,nB,nC) = \text{int}\left( \frac{k}{2nB + nC} \right) + \text{round}\left( \frac{k \% (2nB + nC)}{2nB + nC} + 0.5 \right) \]

where int( ) denotes integer division where the fraction part is discarded, round( ) denotes rounding to the nearest whole number [e.g. round(0.5)=1], and % denotes the modulus operator [in general: a\%b = a - int(a/b) * b].
Step 2: Repeat for each packet to be sent
Step 3: Construct an IP packet with a random type.
// This is important to further confuse and mislead someone who is sniffing the
// communication channel
Step 4: Insert the list of IP addresses (List_IP) in the record route option
Step 5: Repeat for each IP address in the record route option
If there is no more characters in HM, exit.
// the IP address belongs to Class C
Replace the least significant byte by the next character from HM
Else // Class B or Class A
Replace the least significant two bytes by the next two characters from HM
End // repeat step 5
End // repeat step 2
End // Algorithm generatePackets
```

Using this algorithm, the secret information will be hidden inside valid IP addresses to protect the covert channel against steganalysis. However, this would require more packets to be generated since the algorithm uses only one or two bytes in each IP address in the record route options. If further secrecy is deemed important, the confidentiality of the hidden information can be enhanced further using any crypto technique.

### 4.3 Example

Assume that we want to send the Hidden Message (HM) ‘RDV at 9pm’, which has ten characters from a source host (190.100.20.10) to a destination host 195.95.40.10 as shown in Figure 12. The command ‘traceroute’ retrieves the following two IP addresses of Class B and two IP addresses of Class C from the connection path:
A novel covert channel based on the IP header record route option

- Class B addresses:
  190.100.20.1
  190.100.30.1
- Class C addresses:
  195.95.37.1
  195.95.40.1.

Figure 12  An example of a connection path

By applying the algorithm, the number of packets that should be sent to carry HM is computed as follows:

\[ P(10, 2, 2) = \text{int}(10/6) + \text{round}(0.67 + 0.4) = 1 + 1 = 2 \text{ packets}. \]

The contents of the record route options of the two packets that are generated by the algorithm are shown in Figure 13. As shown in Figure 13, the hidden message HM will be sent to the destination host in two separate packets. Packet 1 will carry the string “RDV at” and Packet 2 will carry the remaining string “9pm”. For Class C IP addresses, we modified only the least significant byte, and the least significant two bytes for Class B IP addresses. It is clear from Figure 12 that even if the traffic is sniffed and analysed, it will be very difficult to notice that the IP addresses in the record route options are not valid addresses that carry hidden messages.

Figure 13  The contents of the record route options of the two packets

<table>
<thead>
<tr>
<th>Source host</th>
<th>Path 1</th>
<th>Path 2</th>
<th>Destination host</th>
</tr>
</thead>
<tbody>
<tr>
<td>190.100.20.1</td>
<td>190.100.20.1</td>
<td>190.100.20.1</td>
<td>190.100.20.1</td>
</tr>
<tr>
<td>190.100.20.1</td>
<td>190.100.20.1</td>
<td>190.100.20.1</td>
<td>190.100.20.1</td>
</tr>
</tbody>
</table>

Compared with the contents of the record route option of Figure 9 (which was generated without using the generatePackets algorithm), Figure 13 shows, clearly, the strength of the algorithm in protecting the covert channel against steganalysis. Figure 9 includes non-valid addresses such as 112.109.0.0, which is quite suspicious to be included in any
record route option. On the contrary, all IP addresses appear in Figure 13 look-like valid router IP addresses.

4.4 Covert memory per packet

The proposed covert channel offers more covert memory per packet than the existing available covert channels. The number of bytes in the covert memory depends on the classes of the IP addresses of the routers between the source host and the destination host. The following formula computes the number of bytes \( n \) available in the covert memory per packet:

\[
    n = (2 \times m1) + m2
\]

where

- \( m1 \): the number of Class A and Class B’s IP addresses of the routers between the source host and the destination host
- \( m2 \): the number of Class C’s IP addresses of the routers between the source host and the destination host.

Table 2 gives examples of the number of bytes \( n \) available in the covert memory per packet. Table 3 shows the covert memory per packet in a number of existing covert channels. The proposed covert channel in this paper offers the highest covert memory, when there are more than three routers between the source host and the destination host (Table 2). In addition, it is not limited to any particular protocol type. For example, the packets carrying the hidden information can be a combination of ICMP, TCP and UDP packets.

<table>
<thead>
<tr>
<th>No. of routers between source and destination</th>
<th>No. of class A and B’s IP addresses (m1)</th>
<th>No. of class C’s IP addresses (m2)</th>
<th>No. of bytes in the covert memory per packet ( n ) (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>2</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 3 Available covert memory per packet in some existing covert channels

<table>
<thead>
<tr>
<th>Protocols</th>
<th>The fields carrying the hidden information</th>
<th>Covert memory/packet</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP</td>
<td>The Identification field</td>
<td>2 bytes/packet</td>
</tr>
<tr>
<td>TCP</td>
<td>The Initial Sequence Number (ISN) field</td>
<td>4 bytes/packet</td>
</tr>
<tr>
<td>TCP</td>
<td>The Acknowledge Sequence Number field</td>
<td>4 bytes/packet</td>
</tr>
<tr>
<td>TCP</td>
<td>The TCP Options field (TCP timestamp)</td>
<td>4 bytes/packet</td>
</tr>
<tr>
<td>TCP</td>
<td>TCP flags Bits (URG, ACK, PSH, RST, SYN, FIN)</td>
<td>Few bits/packet</td>
</tr>
<tr>
<td>ICMP</td>
<td>The Data field</td>
<td>4 bytes/packet</td>
</tr>
</tbody>
</table>
5 Implementation

A friendly graphical tool has been developed based on the proposed covert channel, using Visual C++ and Winsock library. At the source host, the tool allows a user to write his hidden message and then generates the necessary packets that will carry the hidden message. At the destination host, the tool extracts the hidden messages from the received packets and displays their contents to the user.

The following steps describe the process of sending an example hidden message to a destination host using the tool. Figure 14 shows the network’s architecture used in this example. The network has two Cisco routers (2600 series) connected via a serial interface. The first router is connected to the source host via subnetwork 2.2.2.x, and the second router is connected to the destination host via subnetwork 1.1.1.x.

**Figure 14** Network architecture

![Network Architecture Diagram]

**Step 1:** At the source host, as soon as the user invokes the tool, he will get the main screen shown in Figure 15. The source IP address of the source host will be displayed automatically by the tool. The user needs only to write the IP address of the destination host (1.1.1.12), and his hidden message (‘Meet you in Tunis’) as shown in Figure 15.

**Step 2:** The user clicks on the ‘Traceroute’ button to get the list of IP addresses of the routers between the source host and the destination host. In response, the tool will automatically execute the ‘traceroute’ command and retrieve the list of IP addresses along the path to destination. Then it computes the minimum number of packets required to send the hidden message, and displays this information, as shown in Figure 16.
Figure 15  Main screen of the covert channel tool

Figure 16  The result of the ‘Traceroute’ command
To compute the minimum number of packets needed to carry the hidden message, it is important to note that, although the number of characters in the hidden message is 17, the tool will use 18 bytes to send the message. The first byte will include the number of character in the hidden message and the remaining 17 bytes will include the ASCII codes of the 17 characters of the hidden message. Since there are three class-A IP addresses identified along the path to the destination, and the tool uses the two least significant bytes of each class-A IP address to carry two characters of the message (see the algorithm `generatePacket`, Section 4.2); then each packet will carry six characters of the hidden message. Consequently, three packets are required to send the hidden message.

**Step 3:** Once the user agrees with the identified list of IP addresses and the number of packets to be used (Figure 16), the tool will automatically generate the three packets with random types, which could be ICMP and/or UDP. The types of the ICMP packets are also chosen randomly, in order to avoid one type of packets carrying the hidden message. This would contribute considerably to further protect the covert channel from steganalysis. Figure 17 shows that two ICMP packets (Type = 15 and Type = 13) and one UDP packet will be generated to carry the hidden message ‘Meet you in Tunis’.

**Step 4:** At the destination host (1.1.1.12), the tool uses a graphical interface to extract and reconstruct the hidden message inserted in the three received packet as shown in Figure 18.

**Figure 17**  The types of the three packets used to send the hidden message
In order to analyse the three packets received at the destination host, the Sniffer CommView is used to capture and display the contents of the packets. The first captured packet is an ICMP packet whose type is 15 (Figure 19). The IP header’s option of the packet is the record route option, and has three IP addresses which include the number of the characters in the hidden message (17) and the ASCII codes of the characters in the string “Meet”, namely:

- The first IP address is 2.2.17.77:
  - 17 is the number of characters in the hidden message
  - 77 is the ASCII code of the first character in the hidden message (‘M’).
- The second IP address: 4.4.101.101:
  - 101 is the ASCII code of the character (‘e’)
  - 101 is the ASCII code of the character (‘e’).
- The third IP address: 1.1.116.32:
  - 116 is the ASCII code of the character (‘t’)
  - 32 is the ASCII code of the space character (‘ ’).
A novel covert channel based on the IP header record route option

Figure 19 The ICMP packet (Type = 15) carries the number of the characters in the hidden message and the string “Meet”

The second captured packet is an ICMP packet whose type is 13 (Figure 20). The IP header’s option is the record route option, and has three IP addresses which include the ASCII codes of the characters in the string “you in”, namely:

- The first IP address is 2.2.121.111:
  - 121 is the ASCII code of the character (‘y’)
  - 111 is the ASCII code of the character (‘o’).

- The second IP address: 4.4.117.32:
  - 117 is the ASCII code of the character (‘u’)
  - 32 is the ASCII code of the space character (‘ ’).

- The third IP address: 1.1.105.110:
  - 105 is the ASCII code of the character (‘i’)
  - 110 is the ASCII code of the character (‘n’).

The third captured packet is an UDP packet (Figure 21). The IP header’s option of the packet is the record route option, and has three IP addresses which include the ASCII codes of the characters in the string “Tunis”, namely:

- The first IP address is 2.2.32.84:
  - 32 is the ASCII code of the space character (‘ ’)
  - 84 is the ASCII code of the character (‘T’)

- The second IP address: 4.4.117.110:
  - 117 is the ASCII code of the character (‘u’)
  - 110 is the ASCII code of the character (‘n’).
The third IP address: 1.1.105.115:

- 105 is the ASCII code of the character ('i')
- 115 is the ASCII code of the character ('s'),

It is clear from Figures 19–21 that the tool successfully generates the required number of packets and embeds the hidden message in the appropriate places in the generated packets.

Figure 20 The ICMP packet (Type = 13) carries the string “you in” of the hidden message

Figure 21 The UDP packet carries the string “Tunis”
6 Conclusion

This paper discusses a novel covert channel for exchanging secret information based on the IP header record route option. The covert channel is protected against steganalysis and sniffing by hiding the secret information inside valid IP addresses in the record route options. An algorithm has been developed to generate the necessary packets to carry the secret information. In order to avoid a single type of packet carrying the hidden message, the types of the packets are chosen randomly, and could be ICMP and/or UDP. This would contribute considerably to further protect the covert channel from steganalysis. Compared to the existing covert channels proposed in the literature, our covert channel offers more covert memory than any of the existing covert channels, especially when there are more than three routers between the source host and the destination host. It also exploits the simplicity of the ICMP and UDP tunnelling to avoid the restriction and rules (synchronisation, flow and congestion control) imposed by TCP-based covert channels.

A friendly graphical tool has been developed to demonstrate the proposed covert channel. The tool allows a user to write his hidden message and then generates the necessary packets that will carry the message. At the destination host, the tool extracts the hidden messages from the received packets and displays their contents to the user. The proposed technique is not suitable to hide multimedia data as this requires significant memory; however, the technique can be used to synchronise the transmission of hidden multimedia data.

Currently, we are developing new mechanisms to further protect the scheme from other advanced steganalysis, especially in networking environments that are highly protected by Firewalls (using filtering rules) and intrusion detection systems.

References


