RESEARCH ARTICLE OPEN ACCESS

Implementation of Covert Channel Method Based on IPv4 Identification Field over NS-3

Hamza Kheddar*, Merouane Bouzid**
*(Department of Telecommunication, LCPTS Lab USTHB University, Algeria)
** (Department of Telecommunication, LCPTS Lab USTHB University, Algeria)

ABSTRACT
Recently, there has been a noticeable increase in the interest in VoIP steganography due to the volume of VoIP traffic generated, which proved to be economically feasible to utilize .In this paper we present first available steganographic techniques that can be used for creating covert channels for each TCP/IP layer in VoIP application. Then we present steganographic methods which hide secret data in IP protocol header fields, particularly the identification field. The IP protocol covert channel implementation was carried out in NS-3 (Network Simulator 3).
Keywords - Data hiding, network steganography, Secure VoIP, TCP/IP protocols, Covert Channel.

I. INTRODUCTION

The covert channel concept was introduced in 1973 [1]. A covert channel is a communication channel that allows two cooperating processes to transfer information in a manner that it does not respect the system's security policy [2]. It means a communication that is not part of the original system which can be used to transfer information to a process or user in a hidden manner.

Covert channels exist only in systems with multilevel security [3], which contain and manage information with different sensitivity levels. It allows different users access to the information, at the same time, with different itineraries.

VoIP is one of the most popular services in IP networks and it stormed into the telecom market and changed it entirely. As it is used worldwide more and more willingly, the traffic volume that it generates is still increasing. That is why VoIP is suitable to enable hidden communication throughout IP networks.

VoIP covert channels have different applications as they can pose a threat to the network communication or may be used to improve the functioning of VoIP (e.g. security as in [4] or quality of service [5]). The first application of the covert channel is more dangerous as it may lead to the confidential information leakage. It is hard to assess what bandwidth of covert channel poses a serious threat. It depends on the security policy that is implemented in the network. For example, US Department of Defense specifies in [6] that any covert channel with bandwidth higher than 100 bps must be considered insecure for average security requirements. Moreover for high security requirements it should not exceed 1 bps.

In this paper we present a study of covert channels that may be applied for the layered TCP/IP model in a VoIP communication. We will focus in particularly, on the implementation of a steganographic method based on hiding data in identification field which located on the IP header (fig.1).

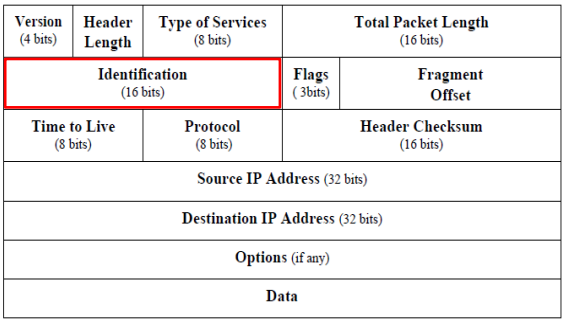

Fig.1 IPv4 Header.

II. TCP/IP MODEL

TCP/IP is a set of network protocols developed for Internet from the 1970s. As a protocol suite based on layers, TCP/IP has a number of weaknesses that allow an attacker to leverage techniques in the form of covert channels to secretly pass data in ordinary packets [7].

The appropriateness of protocol layers for covert channels is evaluated with respect to three criteria: technical difficulty, generality and reachability.

- *Technical Difficulty*: What are the technical requirements and barriers to establish and read a covert channel? Does it require: special hardware, alteration of the operating system, low level programming, developing an application or simple system configuration?

- *Generality:* Once the technical barriers for a covert channel have been overcome, how widely can they be applied? Is all Internet traffic susceptible or only some subset thereof?
- *Reachability:* If a covert channel is established, how far can it reach through the Internet? For example, is it likely to be confined within an institution (LAN) or be on a global scale (WAN).

In this section, a high level description of each layer of the TCP/IP protocol stack is presented. The characteristics and potential for covert channels of each layer is briefly discussed [8], [9]. In the protocol stack each layer is implemented by one or more protocols (horizontal) and one or more interfaces (vertical), as outlined in Fig. 2.

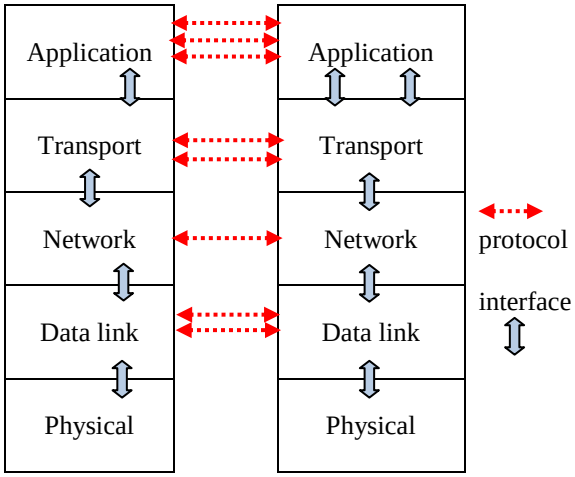

Fig.2 TCP/IP 5-Layer Model.

A. Physical layer

The physical layer contains all the functions needed to carry the bit stream over a physical medium to connect adjacent nodes. The protocols in this layer depend on the actual transmission medium used. There are a number of barriers that make this layer a difficult environment for establishing, encoding and decoding the covert channel. This would probably require special hardware to be designed and installed. In addition, the diversity of protocols and media mean that different technologies would have to be developed for each type of network which yields to limit the generality. Furthermore, the reachability of the covert channel would be often limited to adjacent nodes (within an institution or some subset thereof) with some limited extension possible where repeaters are in use.

B. Link layer

The link layer is responsible for organizing the bit stream into a data unit called a frame and delivery the frames between adjacent nodes on a network. It provides physical addressing (port number), and error detection (such as cyclical redundancy check). The link layer also includes the Media Access Control (MAC) sub-layer, which provides physical addressing and channel access control mechanisms that enable several terminals or network nodes to communicate in a network.

There are a number of technical difficulties that would need to be overcome to establish covert channels at this level. Special hardware or low level device driver programming is necessary, because different types of network such as Ethernet, ATM and Token-Ring have different link layer protocols. The reachability of covert channels at this layer will often be limited to a LAN (within an institution) as long as there are logical and physical limits, with some limited extension possible where level-3 network router are in use.

Two potential ways in which the link layer may be used for covert channels are given on [8], [9]. Firstly, the collision detection system (Carrier Sense Multiple Access / Collision Detection CSMA/CD) in the Ethernet link layer can be modified to transmit hidden data by adjusting the collision control mechanism. Secondly, unused portions of the frame can be used to store covert data. Covert data can be stored in the buffer, beginning at the end toward the valid data. When the packet is transmitted, the entire buffer is exported, including the covert data.

C. Network layer

The network layer delivers data in the form of a packet from source to destination, across as many links as necessary. At the Network layer, the packets of the communication need to be identified with the source and destination addresses of the two end nodes. With IPv4, this means that each packet has a 32-bit source address and a 32-bit destination address in the IP protocol header. The IP protocol defines the addressing system and how intermediary nodes should treat packets. Each packet can be independently routed from source to destination. It provides a best effort service to higher layers. The network layer also contains control and routing protocols.

Covert channels can be created at the network layer; however this is requiring device driver programming or alteration of operating system code. Developing a single technology to establish covert channels in the network layer, would yield a high degree of generality and global reachability.

Analysis of the IP header shows the existence of bits that are either unused or optional. Consequently, fields from the IP header can be manipulated to store covert data. An extended analysis of using the IP identification field as a covert channel is presented later in this paper.

D. Transport layer

Process-to-process delivery is the task of the transport layer. Getting a packet to the destination

system is not quite the same thing as determining which process should receive the packet's content. A system can be running file transfer, email, and other network processes all at the same time, and all over a single physical interface. Naturally, the destination process has to know on which process the sender originated the bits inside the packet in order to reply. There are two main Internet transport protocols: the Universal Datagram Protocol (UDP) and the Transmission Control Protocol (TCP). TCP is a connection-oriented, "reliable" service that provides ordered delivery of packet contents. UDP is a connectionless, "unreliable" service that does not provide ordered delivery of packet contents [10].

The location of transport protocol implementations within the operating system, make the technical difficulties in creating covert channels similar to the Internet layer. The creation of adapted device drivers or the modification of operating system source code is likely to be necessary. It is variously estimated that over 80% all Internet traffic [11] is carried by TCP therefore establishing a covert channel within TCP provides a high level of generality. The fact that TCP is an end-to-end protocol gives it global reachability. In addition, the TCP header is more complex than IP and has a significant array of options, which means there is more potential for the creation of such channels.

Two examples of using TCP fields to covert channels [12], [13] are given below:

1) *Initial Sequence Number Field:* The Initial Sequence Number (ISN) field 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". For our purposes, the sequence number field serves as a good medium for transmitting clandestine data because of its size (a 32 bit number). Hence, there are a number of possible hiding methods to use. The simplest is to generate the sequence number from our actual ASCII character we wish to have encoded.
2) *The TCP Acknowledge Sequence Number Field*: This method, called "*Bounce*", relies upon basic spoofing of IP addresses to enable a sending host to "bounce" a packet of information outside of a remote site and have that site return the packet to the real destination address. This has the benefit of concealing the packet sender 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 number plus one (ISN+1).

E. Application layer

This layer handles issues like network transparency, resource allocation and problem partitioning. The application layer is concerned with the user's view of the network, such as formatting e-mail messages [10]. It contains a diverse range of protocols for applications such as email, remote administration, World Wide Web (WWW) and Peer-to-Peer content distribution. An important application is DNS (Domain Name System) which translates host names into IP addresses.

The technical difficulties in creating covert channels are easiest to overcome at the application layer where limited or no programming skills may be required. The diversity of protocols make the generality of a technical solution limited, however most networked computers can be expected to support common protocols such HTTP and SNMP. So, reachability will be global.

The application layer is nearest the user who can create applications utilising system resources including the network. Many of the classical steganographic approaches can be used at the application level. For example, a covert messaging system can be devised using word substitution in an email system.

For all these types of steganographic (TCP/IP protocols Steganography) methods mentioned on this paper, achieved steganographic bandwidth can be expressed as follows [14]:

$$PRBR_{NS} = \frac{\left(SB_0 + \sum_{j=1}^{l} SB_j\right)}{l+1} \ [bits/packet] \qquad (1)$$

Where:

- $PRBR_{NS}$ (Packet Raw Bit Rate) denotes bandwidth of the covert channel created by TCP/IP steganography [bits/packet].
- SB_0 is the total amount of bits for TCP/IP protocols that can be covertly send in the fields of the first packet.
- SB_j denotes total amount of bits for TCP/IP protocols that can be covertly sent in the fields of the following packets, l is number of packets send besides first packet.

III. IMPLEMENTATION OF COVERT CHANNEL BASED ON IP IDENTIFICATION FIELD

In this section, we present a network steganographic method based on hiding data in identification field which located on the IP header. Our implementation of a VoIP network system was carried out by using the Network Simulator NS-3

[15] to obtain realistic scenario of IP packet transmission.

Fig.3 shows the point-to-point topology used in the simulation. It is a common configuration where traffic is generated from one node of a network and forwarded to other node. This output link is represented by an Ethernet link between the first Ethernet network node and the other side node. Our steganographic method has been implemented on IP header situated exactly on the layer three of IP protocol. It means that we have extended the IP protocol with an algorithm to enable covert channel on identification field.

Notice that our simulation was carried out with traffic sources Constant Bit Rate (CBR) of 5 Mbits and 2 ms for the delay. In our simulation of the covert channel based on IP header identification field, we have transmitted the data word "hello" as a secret message, the actual packet data does not matter. We have tested it in text format which is the easiest way for result exhibition and we can decode it from ASCII format to text format manually. The implementation of this method gives the result shown fig.4.

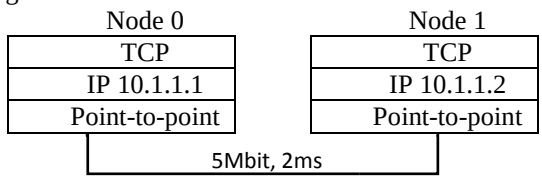

Fig. 3 Topology used in the simulation.

Fig. 4 shows an example of trace file for the captured packet transmitted during the communication between the two nodes. The file can be generated either with **.tr** format which can be read by a simple software like Notepad (software exist on windows), or with **.pcap** format which can be read by Wireshark software [16].As we can see clearly in this figure, the first identification field (id) of the first packet equal **0**, the second id equal 104 which is the ASCII code of the letter **h**, the third id equal 101 which is the ASCII code of the letter **e**, the fourth id equal 108 which is the ASCII code of the letter **l**, the fifth id equal 108 which is the ASCII code of the letter **l** and so on.

Notice that the first identification field (id) of the first packet equal **0** (fig.4), this value (SB_0 in Eq.1) differs from the values achieved for the following packets ($\sum_{j=1}^{l} SB_j$ in Eq. 1) because in the first packet initial values of certain fields can be used (e.g. sequence number for TCP protocol) [14].

In our simulation, we have sent 20 packets, while our secret message "hello" needs 6 packets to be completely transferred (1 packet for each character plus 1 for the first packet), so it remain 14 packets, hence our covert channel will be created for the 6 packets then, our implementation assigns the default value of identification field which NS-3 generate it without covert channel (default configuration) for each remaining packet.

```
th: 40 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ SYN ] Seq=0 Ack=0 Win=65535)
ot-ECT ttl 64 id 0 protocol 6 offset (bytes) 0 flags [none] length: 40 10.1.1.2 > 10.1.1.1) ns3::TcpHeader (8080 > 49153 [ SYN  ACK ] Seq=0
ot-ECT ttl 64 id 0 protocol 6 offset (bytes) 0 flags [none] length: 40 10.1.1.2 > 10.1.1.1) ns3::TcpHeader (8080 > 49153 [ SYN  ACK ] Seq=0
th: 40 10.1.1.2 > 10.1.1.1) ns3::TcpHeader (8080 > 49153 [ SYN  ACK ] Seq=0 Ack=1 Win=65535)
ot-ECT ttl 64 id 104 protocol 6 offset (bytes) 0 flags [none] length: 40 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=1 Ack
ot-ECT ttl 64 id 104 protocol 6 offset (bytes) 0 flags [none] length: 40 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=1 Ack
ot-ECT ttl 64 id 101 protocol 6 offset (bytes) 0 flags [none] length: 576 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=1 Ac
t-ECT ttl 64 id 101 protocol 6 offset (bytes) 0 flags [none] length: 576 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=1 Ack
gth: 40 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=1 Ack=1 Win=65535)
ngth: 576 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=1 Ack=1 Win=65535) Payload Fragment [0:536]
ngth: 40 10.1.1.2 > 10.1.1.1) ns3::TcpHeader (8080 > 49153 [ ACK ] Seq=1 Ack=537 Win=65535)
ot-ECT ttl 64 id 108 protocol 6 offset (bytes) 0 flags [none] length: 576 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=537
ot-ECT ttl 64 id 108 protocol 6 offset (bytes) 0 flags [none] length: 576 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=537
ot-ECT ttl 64 id 108 protocol 6 offset (bytes) 0 flags [none] length: 576 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=1073
ot-ECT ttl 64 id 108 protocol 6 offset (bytes) 0 flags [none] length: 576 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=1073
ngth: 576 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=537 Ack=1 Win=65535) Payload Fragment [536:1040] Payload Fragment [0
ngth: 576 10.1.1.1 > 10.1.1.2) ns3::TcpHeader (49153 > 8080 [ ACK ] Seq=1073 Ack=1 Win=65535) Payload Fragment [32:568]
ot-ECT ttl 64 id 101 protocol 6 offset (bytes) 0 flags [none] length: 40 10.1.1.2 > 10.1.1.1) ns3::TcpHeader (8080 > 49153 [ ACK ] Seq=1 Ack
ot-ECT ttl 64 id 101 protocol 6 offset (bytes) 0 flags [none] length: 40 10.1.1.2 > 10.1.1.1) ns3::TcpHeader (8080 > 49153 [ ACK ] Seq=1 Ack
```

Fig. 4 Trace file of the end-to-end communication scenario.

It is also worth noting that we can represent a 32-bit IPv4 address as a 128-bit IPv6 address by implementing or activating an existing transition mechanism such as dual stack, IPv4 compatible address formats or Tunneling mechanisms[17], thus our method of steganography can pass through an IPv6 network. Consequently the reachability will be much more global than before.

IV. CONCLUSION

In this paper, we presented first an analysis of the potential for covert channels in each layer of the TCP/IP protocol stack. The analysis revealed that the network and transport layers, which they have consecutively the IP and the TCP protocols, are the most vulnerable to the creation of covert channels.

The implementation confirms that the identification field in the IPv4 can be used to store a specific value to be passed to the other end, so it is possible to use this method to pass data between hosts in any IP packet.

Our future work aim is to develop a mechanism to make relationship between the network Steganography (covert channels) based on protocols and the data hiding applied on speech coder. In other word, developing a method to control the covert channels based on protocols via payload coded data and vise versa.

REFERENCES

[1] B. W Lampson. A Note on the Confinement Problem. *USA, Xerox Palo Alto Research Center, 1973.*

[2] S. Berg, Glossary of Computer Security Terms. *USA, National Computer Security Center, 1998.*

[3] N. E. Proctor, P. G. Neumann, Architectural implications of Covert Channels. *USA, Computer Science Lab, SRI International, 1992.*

[4] W. Mazurczyk, Z .Kotulski, New Security and Control Protocol for VoIP Based on Steganography and Digital Watermarking. *In Proc. of: IBIZA 2006, Kazimierz Dolny, Poland, 2006.*

[5] W. Mazurczyk, Z. Kotulski, New VoIP Traffic Security Scheme with Digital Watermarking. *In: J. Górski, SAFECOMP 2006. LNCS, vol. 4166, pp. 170–181. Springer, Heidelberg, 2006.*

[6] US Department of Defense, *Department of Defense Trusted Computer System Evaluation Criteria, DOD 5200.28-STD* (The Orange Book1985).

[7] D. Llamas, *Covert channel analysis and data hiding in the TCP/IP protocol suite*. Honours Project Thesis. UK, Napier University, 2004.

[8] C. G. Girling, Covert Channels in LAN`s. *USA, IEEE Transactions on Software Engineering, 1987*.

[9] T. G. Handeland, M. T. Sandford. Hiding Data in the OSI Network Model. *USA, Weapon Design Technology Group, Los Alamos National Laboratory, 1996.*

[10] D. Howe, "*FOLDOC Computer Dictionary*". USA, Webnox Corp 2003.

[11] S. Lam, Back to the Future Part 4: The Internet. *USA, ACM SIGCOMM Computer Communication Review, 2005.*

[12] Route Project Loki: ICMP Tunnelling. *USA, Phrack Magazine 1996.*

[13] Rowland, C. H. USA,Vol.2 No.5- First Monday Magazine, 1996.

[14] W. Mazurczyk and K. Szczypiorski Steganography of VoIP Streams. *OTM 2008, Part II, LNCS 5332, pp. 1001–1018, Springer- Heidelberg 2008.*

[15] (2014) the ns-3 website. [Online]. Available: www.nsnam.org/

[16] (2014) the wireshark website. [Online]. Available: https://www.wireshark.org/.

[17] Sun Microsystems (2014) System Administration Guide Volume 3, chapter 15"Transition from IPv4 to IPv6" homepage. [Online]. Available: *www.*docs. oracle.com/cd/E19455-01/806 *0916/index.html.*