2009 International Conference on Multimedia Information Networking and Security

Steganography in Handling Oversized IP Packets

Wojciech Mazurczyk and Krzysztof Szczypiorski
Institute of Telecommunications
Warsaw University of Technology
Warsaw, Poland
e-mail: {wmazurczyk, ksz}@tele.pw.edu.pl

***Abstract*—This paper identifies new class of network steganography methods that utilize mechanisms for handling oversized IP packets: IP fragmentation, PMTUD (Path MTU Discovery) and PLPMTUD (Packetization Layer Path MTU Discovery). In particular, for these mechanisms we propose two new steganographic methods and three extensions of existing ones. We present how mentioned mechanisms can be used to enable hidden communication for both versions of IP protocol: 4 and 6. Also the detection of the proposed methods is enclosed in this paper.**

Keywords: steganography, fragmentation, PMTUD, PLPMTUD

I. INTRODUCTION

Steganographic methods hide secret data in users' normal data transmissions and in ideal situation hidden information and existence of hidden communication cannot be detected by third parties. Various steganographic methods have been proposed and analyzed, e.g. [1]-[4]. They may be seen as a threat to network security as they may be used as a tool to cause for example confidential information leakage. That is why it is important to identify potential possibilities for covert communication, because knowledge of the information hiding procedure can be used to develop countermeasures.

Both versions of IP protocol 4 [5] and 6 [9] were designed to be used on various transmission links. The maximum length of an IP packet is 64 kB but on most transmission links maximum packet length is smaller. This limited value characteristic for the specific link is called a MTU (Maximum Transmission Unit). MTU depends on the type of the transmission link e.g. for Ethernet – 1500, wireless IEEE 802.11 – 2300 and PPP (Point to Point Protocol) – 296 bytes.

There are two possibilities to transmit large IP packet through an end-to-end path that consists of links with different MTUs:

- Permit to divide oversized packet to smaller ones with IP fragmentation [5] mechanism.
- Do not allow packet fragmentation and adjust IP packet size to so called PMTU (Path MTU) – the smallest, acceptable MTU along the entire end-to-end path. For this purpose two methods have been proposed PMTUD for IPv4 [6] and for IPv6 [7] and PLPMTUD [8], which is enhancement of previous method for both versions of IP protocol.

Mechanisms for handling oversized packets like IP fragmentation, PMTUD or PLPMTUD are needed and used in network scenarios where in the end-to-end path intermediate links have smaller MTUs than the MTU of the end links. Below typical network scenarios that require dealing with oversized packets are listed:

- Usage of various tunneling protocols like GRE (Generic Routing Encapsulation), IPSec (IP Security), and L2TP (Layer Two Tunneling Protocol) which add headers and trailers thus reduce effective MTU.
- Using PPPoE (Point to Point Protocol over Ethernet) with ADSL (Asymmetric Digital Subscriber Line). PPPoE has 8 bytes header thus it reduces the effective MTU of the Ethernet to 1492 bytes.
- Using MPLS over Ethernet.
- Connections between endpoints in Token Ring or FDDI networks, which have an Ethernet link between them (with lower MTU) and other similar cases.

The objectives of this paper are to:

- Describe mechanisms used to handle oversized packets in IPv4 and IPv6 networks.
- Present exiting network steganography methods that utilize these mechanisms.
- Propose two new steganographic methods and three extensions of existing ones. All presented steganographic methods may be applied to both versions of IP protocol (4 and 6).

The rest of the paper is as follows. Section 2 describes existing mechanisms for handling oversized packets for IPv4 and IPv6 protocols. In Section 3 existing network steganography methods that utilize these mechanism are presented. Section 4 includes detailed description of new information hiding methods and their potential detection. Section 5 concludes our work.

II. OVERVIEW OF MECHANISM FOR HANDLING OVERSIZED IP PACKETS

A. IP Fragmentation

To accommodate MTU differences on links in end-to-end path in IP fragmentation, intermediate nodes are allowed to fragment oversized packets to smaller ones. Then receiver or some other network node (e.g. router) is responsible for reassembling the fragments back into the original IP packet.

978-0-7695-3843-3/09 $26.00 © 2009 IEEE
DOI 10.1109/MINES.2009.246

IP fragmentation mechanism involves using the following fields of the IPv4 header: *Identification*, *Fragment Offset* fields, along with the MF (More Fragments) and DF (Don't fragment) flags. It also needs to adjust values in *Total Length* and *Header Checksum* fields for each fragment to represent correct values. The above header fields are used as follows:

- *Identification* (16 bits) is a value assigned by the sender to each IP packet to enable correct reassembling of the fragments (each fragment has the same *Identification* value).
- *Fragment Offset* (13 bits) indicates which part of the original packet fragment carries.
- *Flags* field (3 bits) contains control flags. Bit '0' is reserved and is always set to 0. Bit '1' is the DF flag – if set to 0 fragmentation can occur; if set to 1 fragmentation is not possible. Bit '2' is the MF flag – if set to 0 and *Fragment Offset* is different from 0, this denotes the presence of last fragment and if set to 1 more fragments are expected to be received.

Similar mechanism is used in version 6 of IP protocol, where *Fragment* extension header is used to perform fragmentation. What differs IPv6 from IPv4 fragmentation is that it may only be performed by the sender and reassembly process have to take place only in the receiver and not in some intermediate node.

TABLE I. IP FRAGMENTATION EXAMPLE

Original IP packet					
Sequence	***Identifier***	***Total Length***	***DF***	***MF***	***Fragment Offset***
0	345	5140	0	0	0

IP Fragments					
Sequence	***Identifier***	***Total Length***	***DF***	***MF***	***Fragment Offset***
0-0	345	1500	0	1	0
0-1	345	1500	0	1	185
0-2	345	1500	0	1	370
0-3	345	700	0	0	555

The example of the IP packet fragmentation for IPv4 is presented in Tab. 1. Original packet which size is 5140 bytes is divided into four fragments of maximum 1500 bytes.

B. PMTUD (Path MTU Discovery)

PMTUD was standardized for IPv4 and published in 1990, but it did not become widely deployed for the next few years. Currently PMTUD is implemented in major operating systems (Windows, Unix, Linux) – in 2002 about 80% - 90% of endpoints on the Internet were using it. As mentioned in the introduction this mechanism was developed to avoid fragmentation in the path between the endpoints. Similar to IPv4 PMTUD mechanism was also developed and standardized for IPv6 [7].

PMTUD is used to dynamically determine the lowest MTU along the end-to-end path between packets sender and receiver. Instead of fragmenting packet, an endpoint determines the largest possible size of the packet that can be sent to a specific destination. An endpoint establishes the correct packet size associated with a specific path by sending packets with different sizes. Packets used by PTMUD are called probe messages and they have DF flag set in the IP protocol header. Their size is initially set to the senders link MTU. While sender generates probes he/she responds to possible ICMP (Internet Control Message Protocol) error reports that indicate a low MTU is present along the connection path. Sender receives a notification informing what packet size will be suitable. The notifications are requested by setting the DF flag in outgoing packets. For IPv4 the notifications arrive as ICMP messages known as "Fragmentation required, and DF flag set" (ICMP type 3, code 4), for IPv6 it is "Packet too big" message from ICMPv6 protocol [10]. PMTUD is working continually during connection because the path between sender and receiver can changed (e.g. because of link failure).

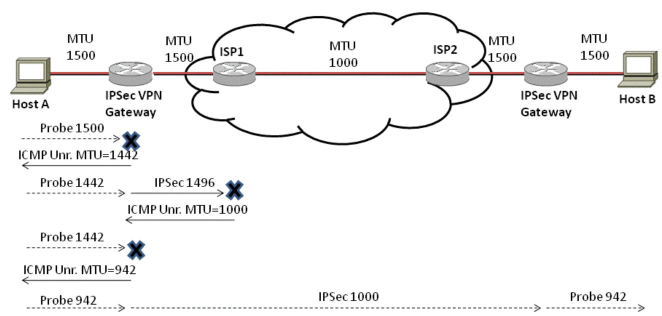

Figure 1. PMTUD example

The PMTUD example is illustrated in Fig. 1. Host A sends packet to host B which size is set to 1500 bytes (default Ethernet MTU). The packet will be transmitted with use of IPSec tunnel, which begins at first router. Because the next link MTU is also 1500 bytes and IPSec adds 54 bytes overhead then total packet size exceeds admissible MTU. Thus the packet is dropped and ICMP message is sent back to the host A with suitable MTU for the next link. Then host A retries sending the packet by reducing its size to 1442 bytes to meet the limit, so packet can successfully traverse through first router. However, the link after next router has MTU of 1000 bytes so the packet is once again dropped and ICMP message is sent in host A direction but it is filtered out by first router. After the timeout expires host A retransmits the packet and receives ICMP message which indicates necessity to decrease packet size to 942 bytes. This last MTU value is then used to successfully exchange data with host B.

It must be noted that there are security issues related with using PMTUD. In particular, sometimes network administrators treat all ICMP traffic as dangerous and block it, disabling possibility of using path MTU discovery. Other potential issues for TCP protocol are described in [11].

C. PLPMTUD (Packetization Layer Path MTU Discovery)

To alleviate issues related with using ICMP traffic for PMTUD described above, enhancement called PLPMTUD was developed and standardized [8]. What differs PLPMTUD from PMTUD is that receiving probes messages are validated at the transport layer. It does not rely on ICMP or other messages from the network, instead it learns about correct MTU by starting with packets which size is relatively small and when they get through with progressively larger ones. In particular, PLPMTUD uses a searching technique to determine optimal PMTU. Each probe narrows the MTU search range. It may raise the lower limit on a successful probe receipt or lower the upper limit if probe fails. The isolated loss of a probe message is treated as an indication of an MTU limit and transport layer protocol is permitted to retransmit any missing data.

III. RELATED WORK

To authors best knowledge, there are no steganographic methods proposed for PMTUD and PLPMTUD mechanisms.

For IPv4 there are few existing methods that utilize IP fragmentation mechanism and fields in IP header related to it. Rowland [1] proposed multiplying each byte of the hidden data by 256 and inserts it directly into *Identification* header field. Cauich et al. [14] described how to use *Identification* and *Fragment Offset* fields to carry hidden data between intermediate nodes but under condition that the packet is not fragmented. Additionally, in selected packet reserved flag is used to mark packet so that the receiver can distinguish between real and covert fragments. Murdoch et al. [4] proposed transmitting hidden information by modulating the size of the fragments to match the hidden data inserted into *Fragment Offset* field. Ahsan and Kundur [12] proposed steganographic method that use IP fragmentation fields. It utilizes high eight bits of the *Identification* to transmit covert data and the low eight bits are generated randomly. The same authors in [13] described a method that uses DF flag as a covert data carrier. If the sender knows the correct MTU for the end-to-end path to the receiver and issues packets which size is less than MTU then DF can be set to arbitrary values.

For IPv6 protocol Lucena et al. [15] identified four network steganographic methods based on *Fragment* header extension. Two methods use reserved fields to carry steganogram and one next header field. Fourth steganographic method is based on fake fragments insertion. In this case all fields of the fragment header may be used for covert communication. To avoid having inserted fragment included in the reassembly process of the original IP packet, authors propose two solutions: first is based on inserting an invalid value in *Identification* field in *Fragment extension* header, thus the receiver will drop such fragment, second – inserting overlapping *Fragment Offset* value that causes data to be overwritten during reassembly. Fake fragments carry hidden data only in certain header fields.

IV. PROPOSED METHODS: COMMUNICATION SCENARIOS, FUNCTIONING AND DETECTION

Every steganographic method should be analyzed in terms of steganographic bandwidth and risk of hidden communication disclosure. Steganographic bandwidth may be expressed by means of RBR (Raw Bit Rate), which is defined as a total number of steganogram bits transmitted during one time unit [bit/s] or equivalently by PRBR (Packet Raw Bit Rate) which is defined as a total number of steganogram bits transmitted in single packet used during the hidden communication process [bit/packet]. Some steganographic methods are trivial to detect (e.g. those which simply modifies header fields) but for others the steganalysis may be harder to perform. Thus, for each proposed steganographic solution potential detection methods must be analyzed.

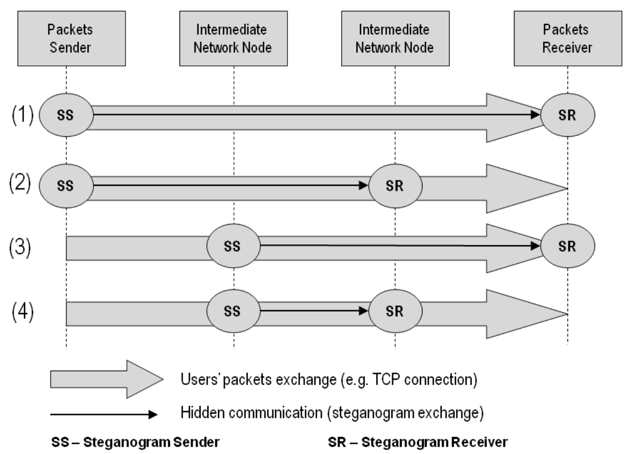

Figure 2. Hidden communication scenarios

In general, there are four communication scenarios possible for network steganographic exchange. The first scenario (1) in Fig. 2, is most common: the sender, who is also a Steganogram Sender (SS) and the receiver, who is also a Steganogram Receiver (SR) establish a connection while simultaneously exchanging steganograms. In the next three scenarios (marked 2-4 in Fig. 2) only a part of the end-to-end path is used for hidden communication as a result of actions undertaken by intermediate nodes; the sender and/or receiver are, in principle, unaware of the steganographic data exchange.

Hidden communication scenarios presented above differ in steganalysis, in particular, the scenario 4 is harder to detect, because the network node which analyses traffic for hidden communication called warden [16] is usually placed at the edge of source or destination endpoints (sub)network.

A. IP fragmentation

For IP fragmentation mechanism we propose new steganographic method (F1) and two enhancement of the previously proposed ones (F2 and F3).

Each of presented methods may be utilized for IPv4 and IPv6 protocols for each scenario from Fig. 2. However, for

IPv4 fragmentation, fragments reassembly may be performed by intermediate nodes as well as by the sender and/or receiver. This may limit the steganogram exchange only to the fragmenting and assembling nodes. For IPv6 there is no such limitation.

Steganographic method F1

In this method SS (Steganogram Sender) must be the source of the fragmentation. SS inserts single bit of hidden data by dividing original IP packet into the predefined number of fragments. For example, if the number of fragments is even then it means that binary "0" is transmitted and in other case binary "1". After reception of the fragments SR uses the number of the fragments of each received IP packet to determine what hidden data was sent.

Potential steganographic bandwidth for this method is PRBR = 1 bit/packet.

Detection of this method may be hard to perform. Statistical steganalysis based on number of fragments can be performed to detect irregularities in number of the fragments. The best method to make hidden communication unavailable is to reassembly original IP packet in the intermediate node responsible for detecting steganographic communication (warden [16]), then refragment it randomly and send to the receiver.

Steganographic method F2

The main idea of this method is to divide a packet into fragments and insert hidden information by modulating the values that are inserted into *Fragment Offset* field. As mentioned in Section 3, Murdoch et al. [4] proposed inserting steganogram directly into *Fragment Offset* field and modulate the size of the fragment to match this value. Such approach can cause high irregularities in fragments sizes which may be easily detected. We propose enhancement of this method which has lower steganographic bandwidth but is harder to detect.

F2 method works as follows. SS must be the source of the fragmentation. SS inserts single bit of hidden data by intentionally modulating the size of each fragment of the original packet in order to obtain fixed values in *Fragment Offset* field. For example, even offset means transmitting binary "1", odd offset – binary "0". "Steganographic" fragmentation of the exemplary IP packet which was introduced in Tab. 1 is presented in Tab. 2. After successful reception of the fragments SR extracts hidden data based on the values from *Fragment Offset* field.

Steganographic bandwidth for this method is $PRBR = N_F - 1$ [bit/packet], where N_F denotes number of packet fragments.

Steganalysis in case of F2 is harder than in case of method proposed by Murdoch, but hidden communication still can be uncovered, because usually all the fragments except last one have equal sizes (see Tab. 1). Thus, if there are any irregularities in fragments sizes, then steganographic communication may be uncovered.

However, this method may be further improved, so the detection is more difficult to perform. We may influence the size of the fragments in such a manner that all fragments except last one would have the same length and the value in *Fragment Offset* field in last fragment is modulated to achieve even or odd value. In this case the hidden communication may not be detected at all. The best method to make the hidden communication unavailable is the same as in case of method F1.

TABLE II. F2 STEGANOGRAPHIC METHOD EXAMPLE

IP Fragments						
Seq.	***Identifier***	***Total Length***	***DF***	***MF***	***Fragment Offset***	***Hidden data***
0-0	345	1300	0	1	0	-
0-1	345	1340	0	1	160	1
0-2	345	1340	0	1	325	0
0-3	345	1220	0	0	490	1

Steganographic bandwidth for this improved method will be lower than for above method and will be equal PRBR = 1 bit/packet.

Steganographic method F3

Proposed method is enhancement of Lucena et al. [15] work for IPv6 fragmentation where they proposed to generate fake fragments. As mentioned in Section 3 two solutions to distinguish fake fragments from the legitimate were presented – first is based on inserting an invalid value in *Identification* field in *Fragment extension* header, second – inserting overlapping *Fragment Offset* value that causes data to be overwritten during reassembly. Fake fragments carry hidden data only in certain header fields. However, described methods may be easy to uncover because the warden can monitor all the fragments sent and determine potential anomalies like overlapping offsets or single, unrelated fragments. Our proposition is to use legitimate fragment with steganogram inserted into payload for higher steganographic bandwidth and harder detection.

F3 method works as follows. SS must be the source of the fragmentation. SS while dividing the packet, inserts steganogram instead of inserting user data into the payload of selected fragment. The problem with such approach is to properly mark fragments used for hidden communication so the receiver can extract it in a way that will not interfere with reassembly process. We propose the following procedure to make the selected fragments distinguishable from others yet hard to detect. Let us assume that sender and receiver share secret Steg-Key (*SK*). For each fragment chosen for steganographic communication the following hash function (*H*) is used to calculate Identifying Sequence (*IS*):

$$IS = H(SK \parallel \mathit{Fragment\ Offset} \parallel \mathit{Identification}) \quad (1)$$

where *Fragment Offset* and *Identification* denote values from these IP fragment header fields and || bits concatenation function. For every fragment used for hidden communications the resulting *IS* will have different value due to the changing values in *Fragment Offset*. All *IS* bits or only selected ones are distributed across payload field in predefined manner. Thus, for each fragment the receiver based on *SK* and values from the IP header can calculate appropriate *IS* and checks if it contains steganogram or user data. If the verification is successful then the rest of the payload is considered as hidden data and extracted. Then SR does not utilize this fragment in reassembly process of original IP packet.

Steganographic bandwidth for this method may be expressed as

$$PRBR = N_F \cdot F_S \quad [bits/packet] \qquad (2)$$

where N_F denotes number of fragments and F_S the size of the fragment payload.

Method F3 is hard to detect because legitimate fragments are used as hidden data carriers. The best method to make the hidden communication unavailable is the same as in case of methods F1 and F2.

B. PMTUD

The main idea for exchanging hidden data with PMTUD is simple – it involves sender to utilize probe messages to carry steganogram and invoke sending intentional fake ICMP messages by receiver. Detailed hidden information procedure is suitable for both IPv4 and IPv6 and is possible for all scenarios from Fig. 2.

Proposed steganographic method works as follows. SS knows from previous interactions with SR what the correct MTU for their communication path is. When SS wants to send steganogram then it sends a probe message that contains steganogram inserted into packet payload. The size of the packet is set to the maximum MTU allowed for path between SS and SR, thus SS is certain that this packet will reach the receiver.

To make the selected packet for steganographic purposes distinguishable from other yet hard to detect we propose similar procedure as it was presented for IP fragmentation mechanism. If we assume that sender and receiver share secret Steg-Key (*SK*), then for each packet chosen for hidden communication a hash function (*H*) is used to calculate Identifying Sequence (*IS*):

$$IS = H(SK \,||\, Identification \,||\, CB) \qquad (3)$$

where *Identification* denotes values from that IP header field, *CB* is *Control Bit* and || is bits concatenation function. *Control Bit* is used to inform the receiver whether it should sent more fake ICMP messages or not (*CB=1* send more ICMP, *CB=0* do not send more ICMP). For every IP packet used for hidden communications the resulting *IS* will be different due to the changing values from *Identification* field. All *IS* bits or only selected ones are distributed across payload field in predefined manner.

After a probe message reaches the receiver, he/she calculates two *IS*s (one for *CB=1,* second for *CB=0*) based on *SK* and value from the IP header and checks if it contains steganogram or user data. When steganogram is detected it is extracted from the packet payload. If *IS* calculation indicates that *CB=1* then receiver intentionally send ICMP message that indicate that the MTU of the path must be decreased and thus sender is obligated to send smaller probe message (which will also contain steganogram). In fake ICMP message source IP address must be spoofed to avoid trivial detection. In the payload of ICMP message IP header of the original packet and 64 bits of original data are present. Receiver must mark ICMP message to allow sender to distinguish real ICMP from fake one. To achieve this we propose to modify the TTL (Time To Live) field of the original IP packet header from the ICMP payload and change the *Total Length* and *Header Checksum* values accordingly. TTL is the only field in IP header (if IP fragmentation is not used) which may be modified during traversing the network. Thus comparing original packet sent with returned in ICMP message will not result in easy hidden communication detection. There are many possibilities of TTL modifications and, in particular, they include setting TTL to prearranged value or to even/odd one. Functioning of the described above steganographic method is also illustrated in Fig. 3. In this example, during the PMTUD exchange, about 3 kB of steganogram was sent from SS to SR.

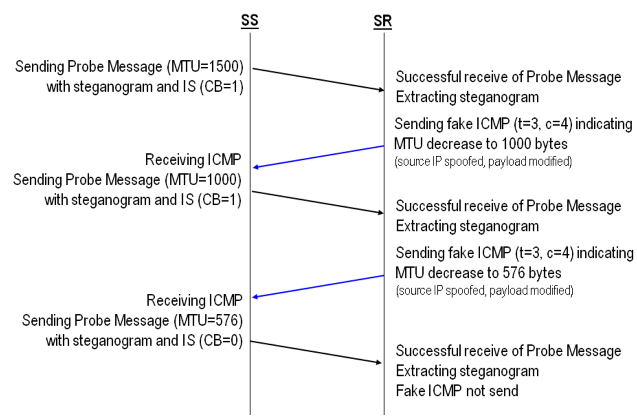

Figure 3. PMTUD steganographic method example

For proposed method steganographic bandwidth can be expressed with as:

$$RBR_{PMTUD} = \frac{\sum_1^n P_n}{T} \quad [bits/s] \qquad (4)$$

where *n* denotes number of probes sent from sender to receiver, P_n probe payload size and *T* connection duration.

During PMTUD exchange all probes messages may be used for steganographic purposes but in this case detection may be easier to perform. Because it is assumed that the earlier probes failed to reach the receiver, next ones should carry fragment of the same data. Thus, comparing each probe message sent with the first one issued may be used to detect steganograms. Only in case when the first probe is used to carry steganogram above steganographic method is hard to detect but then the steganographic bandwidth is limited.

C. PLPMTUD

In PLPMTUD probes messages are validated at the transport layer and correct MTU is learned by starting with packets which size is relatively small and when they get through they proceed with progressively larger ones. The isolated loss of a probe packet is treated as an indication of an MTU limit and transport layer protocol is permitted to retransmit any missing data. Thus, steganographic method described for PMTUD is not applicable. Nevertheless, other possibilities for hidden communication may be utilized. One of them is RSTEG (Retransmission Steganography) method which is presented by authors in details in [17] and uses intentional retransmissions to sent steganograms. RSTEG main idea is to not acknowledge a successfully received packet in order to intentionally invoke retransmission. The retransmitted packet carries a steganogram instead of user data in the payload field. RSTEG may be used for IPv4 and IPv6 in all hidden communication scenarios from Fig. 2.

For PLPMTUD using RSTEG works as follows. SS knows from previous interactions with SR what the correct MTU for their communication path is. When the connection starts, SS sends probe message with prearranged MTU. After successfully receiving the packet, the receiver intentionally does not issue an acknowledgment message. In a normal situation, a sender is obligated to retransmit the lost packet when the timeframe within which packet acknowledgement should have been received expires. In the context of RSTEG, a sender replaces original payload with a steganogram instead of sending the same packet again. When the retransmitted packet reaches the receiver, he/she can then extract hidden information.

The detection method is similar to one presented for PMTUD and is based on comparing probes messages payload during MTU learning process.

V. CONCLUSIONS

In this paper we presented potential steganographic methods that can be used for mechanisms for handling oversized IP packets: IP fragmentation, PMTUD and PLPMTUD. In particular, we propose two new steganographic methods and three extensions of existing ones.

Proposed methods can be utilized to enable hidden communication for both versions of IP protocol: 4 and 6. They are characterized by different steganographic bandwidth and detection possibilities, thus they can have various impact on network security. Knowledge of these information hiding procedures can now be utilized to develop and implement countermeasures for network traffic monitoring. This may limit the risk of confidential information leakage or other threats caused by covert communication.

REFERENCES

[1] C. Rowland, “Covert Channels in the TCP/IP Protocol Suite”, First Monday, Peer Reviewed Journal on the Internet, July 1997

[2] S. Zander, G. Armitage, P. Branch, “A Survey of Covert Channels and Countermeasures in Computer Network Protocols”, IEEE Communications Surveys & Tutorials, 3rd Quarter 2007, Volume: 9, Issue: 3, pp. 44-57, ISSN: 1553-877X

[3] Petitcolas F., Anderson R., Kuhn M., Information Hiding – A Survey: IEEE Special Issue on Protection of Multimedia Content, July 1999

[4] Murdoch S.J., Lewis S., *Embedding Covert Channels into TCP/IP,* Information Hiding (2005), pp. 247-26

[5] Postel J.: *Internet Protocol*, IETF RFC 791, September 1981

[6] Mogul J., Deering S., *Path MTU Discovery*, IETF RFC 1191, November 1990

[7] McCann J., Mogul J., Deering S, *Path MTU Discovery for IP version 6*, IETF RFC 1981, August 1996

[8] Mathis M., Heffner J., *Packetization Layer Path MTU Discovery*, IETF RFC 4821, March 2007

[9] Deering, S., Hinden, R.: *Internet Protocol, Version 6 (IPv6) Specification*, IETF RFC 2460, December 1998

[10] Conta, A., S. Deering, Gupta, M.: Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification, IETF RFC 4443, March 2006.

[11] Lahey, K.: *TCP Problems with Path MTU Discovery*, IETF RFC 2923, September 2000.

[12] Ahsan, K. and Kundur, D.: *Practical Data Hiding in TCP/IP*, Proc. ACM Wksp. Multimedia Security, December 2002.

[13] Kundur D. and Ahsan K.: *Practical Internet Steganography: Data Hiding in IP*, Proc. Texas Wksp. Security of Information Systems, April 2003.

[14] Cauich, E., Gomez Cardenas R., Watanabe, R.: *Data Hiding in Identification and Offset IP Fields*, Proc. 5th Int’l. School and Symp. Advanced Distributed Systems (ISSADS), January 2005, pp. 118–25.

[15] Lucena, N. B., Lewandowski, G., Chapin, S. J.: *Covert Channels in IPv6*, Proc. Privacy Enhancing Technologies (PET), May 2005, pp. 147–66

[16] Fisk, G., Fisk, M., Papadopoulos, C., Neil, J.: *Eliminating Steganography in Internet Traffic with Active Wardens*, In Proc: 5th International Workshop on Information Hiding, Lecture Notes in Computer Science: 2578, 2002, str. 18–35

[17] Mazurczyk W., Smolarczyk S., Szczypiorski K.: *Hiding Information in Retransmissions*, In: Computing Research Repository (CoRR), abs/0905.0363, arXiv.org E-print Archive, Cornell University, Ithaca, NY (USA), May 2009 URL: http://arxiv.org/abs/0905.0363