

About software implementation of network steganography retransmission based algorithm

D. Pavlin¹, A. Makosiy²

¹Siberian State Aerospace University, Institute of Informatics and Telecommunications/ Faculty of Informatics and Computer Science. Krasnoyarsk, Russia

²Khakas State University, Republic of Khakassia, Abakan, Russia

Email address:

Dm1tr1yPafu1n@gmail.com (D. Pavlin), AIMakosi@gmail.com (A. Makosiy)

To cite this article:

D. Pavlin, A. Makosiy. About Software Implementation of Network Steganography Retransmission Based Algorithm. *American Journal of Networks and Communications*. Vol. 4, No. 1, 2015, pp. 1-4. doi: 10.11648/j.ajnc.20150401.11

Abstract: Hybrid network steganography method RSTEG was implemented for Windows TCP/IP stack. The implementation description is provided, including further development prospects.

Keywords: Steganography, Retransmission Mechanism, RSTEG Algorithm

1. Introduction

Modern ways of data transmission and representation, having obvious advantages (ease of recovery, data integrity, prospects of use of universal machine and software solutions) can be crossed out with ease with which data theft and information modification are possible. Therefore there is a logical interest in problems of the development of methods to protect information, primarily in methods of cryptography and steganography. Given the rapid growth of Internet communications, which began to substitute all others, there is a particular interest in methods of secure data transmission over the Internet stream.

It is known that using steganography methods, you can ensure the privacy of transmitted information, by hiding the fact of transfer. A strong argument in favor of usage steganography includes restrictions on the use of cryptographic tools and research in this field in a number of countries. However, it should be noted that steganography cannot provide data integrity in contrast to cryptography.

At present, we may talk about three areas of steganography: classical, computer and digital. Network steganography is on a joint of a computer and digital steganography. It uses computer industry technologies, where data containers (information structure designed to hide a secret message) are digital objects (network packets). Network steganography uses a number of approaches presented on Fig. 1.

Each of network steganography methods use features of a network protocols, or payload fields for transferring hidden

information. Hybrid methods use, both features of protocols, and network packets.

The purpose of this work is an implementation of hybrid steganographic method of data transfer called RSTEG (Retransmission Steganography). The method was proposed by Krzysztof Szczypiorski and his colleagues [1], they also presented simulation of this method in NET2 Simulator.

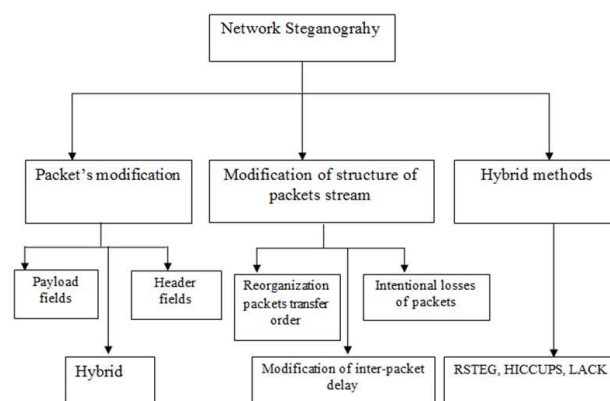


Figure 1. Network steganography methods, classification

2. RSTEG Algorithm

For understanding further material it is necessary to give a general idea of RSTEG. The method is based on network transport layer protocol features TCP, its ability to secure data transfer, more precisely, on the mechanism of packets relay [2]. Next, we will concentrate our attention to relay mechanism on

the base of retransmission timer relay RTO (Retransmission Timeout).

It all begins with sender transferring a segment with legitimate data to receiver (Fig. 2). After reception, the receiver according to the logic of TCP protocol must answer with ACK segment. But receiver doesn't do it intentionally which results in re-sending segment after timeout.

When timeout expires on the sender side, the segment is sent again, but in this case, before sending it, sender modifies payload by embedding steganogram. After that segment is sent to the receiver, which deliberately did not respond with ACK segment.

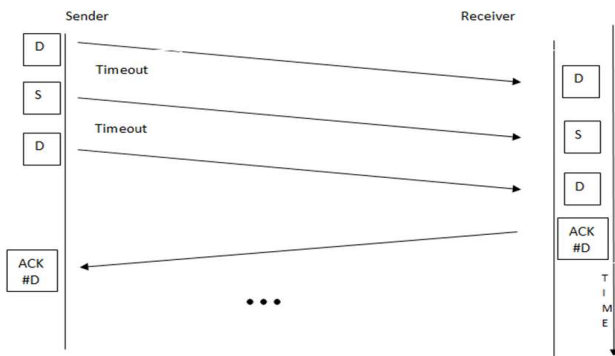


Figure 2. General algorithm of RSTEG

Each of network steganography methods use features of a network protocols, or payload fields for transferring hidden information. Hybrid methods use, both features of protocols, and network packets.

The purpose of this work is an implementation of hybrid steganographic method of data transfer called RSTEG (Retransmission Steganography). The method was proposed by Krzysztof Szczypiorski and his colleagues [1], they also presented simulation of this method in NET2 Simulator.

3. Software Implementation

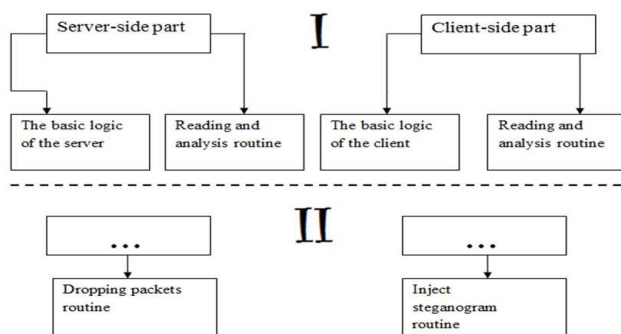


Figure 3. Application design

For the implementation of RSTEG a two-tier client-server application has been developed (Fig. 3).

Each part of the application is working with different levels of abstractions. This resulted in choosing of different programming languages. For the first and second level C# and C were chosen, respectively.

Free open-source library SharpPcap was used for parsing and analysis of network packets on the first level of the application [3]. Another free open-source library WinDivert [4] was used at the second level of the application. This library allows applications running in user mode, to capture, modify and drop network packets from the Windows network stack.

The first level application provides mechanism of message exchange between sender and receiver (both legitimate and hidden data). This level is responsible for establishing a connection and sending messages over a network. Description of RSTEG algorithm doesn't specify exactly when receiver doesn't need to respond. Here it is solved by introduction of a secret key in payload segment – sender and receiver should establish some key (sequence of bytes) in advance which can change from a session to session.

When sending a message, its payload is analyzed by the routine of reading and analyzing of packets on presence of a key at it. Analysis occurs on both client and server side.

To increase adaptability and stegfirmness of the method - subroutine of dropping packets uses a random number from one to four which is responsible for quantity "not answered" segments.

Detailed application diagram is presented on Fig. 4. Ovals represent separate processes. Functions (methods) are italicized, for greater visibility; their names do not accurately match software implementation. Also, along with functions in parentheses it is specified within what process they are executed. Transmitted data is specified above arrows.

The communication session starts with a connection using sockets that are bound to specific ports. Port may vary from session to session; it is passed as a parameter. After the client is connected to the server, subprograms of network packets analysis are started on two ends.

Network packet analysis routines analyze byte sequence payload fields for key segments. For this purpose one of the most productive methods by Hafthor Stefansson is used.

After the connection is established, the exchange of messages begins (step 1, 2). We assume that legitimate data is transferred. It should be noted that at this time, the routine of reading and analysis of packets already works and examines the payload of each incoming packet. At some point (step 3), the sender decides to transfer a key. The packet that contains it is parsed on the client and server, and if successful, subroutines of steganogram injection and drop packets are started. IP address and port are passed as input parameters (step 4).

Then client sends a message. Drop process is running on the server side, it receives the message, but doesn't answer. Timer expires on the client side, and it initiates repeated transfer of a segment. Here, the steganogram injection subprogram is also started. After the expiration of the timer and before sending the message to a network, a segment gets to this subprogram where its payload changes, and then the segment goes down the stack and gets to a network. Then the steganogram injection subroutine stops the work. The server receives a packet with steganogram and doesn't answer again, that involves the third transfer – containing legitimate data.

Segments with steganogram demonstrated below (Fig. 5, 6). Figures show two data link layer frames. The first of these contains a segment with legitimate data. The second frame includes a segment containing steganogram in the payload field. Segment of transport layer starts with two highlighted

contiguous bytes.

It is noticeable that segments are identical, except for a payload and checksum field. Source code for the RSTEG method is available at <https://github.com/Pav1un/RSTEG>.

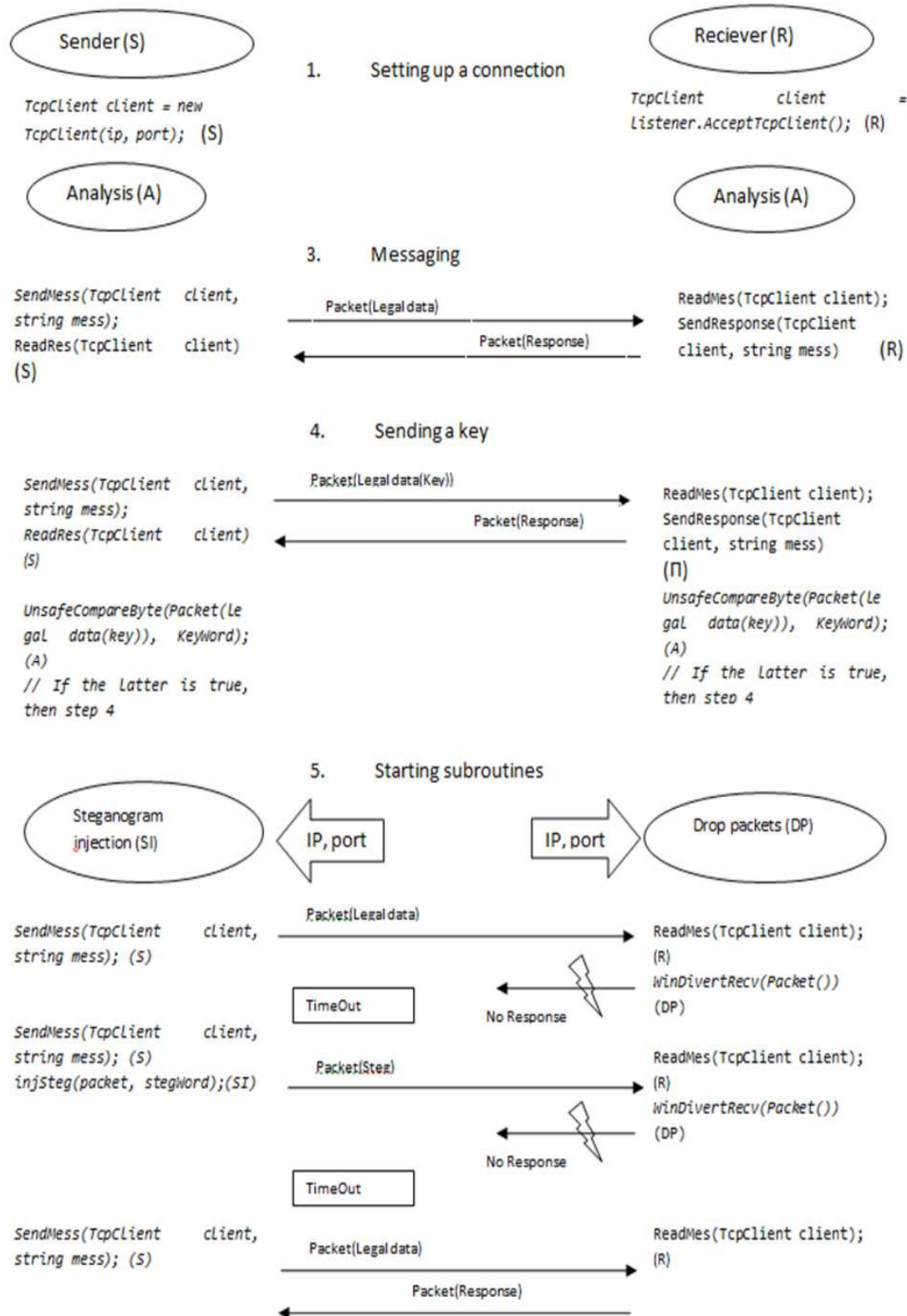


Figure 4. Algorithm from the program point of view

```

54 53 ed 3a 92 d6 00 18 f3 02 4c 97 08 00 45 00 TS.:.... ..L...E.
00 2e 01 eb 40 00 80 06 00 00 c0 a8 00 01 c0 a8 ....@... ..
00 02 0a 8e 06 fa 33 d8 dc b6 51 89 f5 75 50 18 ..3. ..Q...uP.
40 24 81 74 00 00 71 00 77 00 65 00 @$t..q. w.e.

```

Figure 5. Frame with legitimate data

```

54 53 ed 3a 92 d6 00 18 f3 02 4c 97 08 00 45 00 TS.:.... ..L...E.
00 2e 01 e9 40 00 80 06 00 00 c0 a8 00 01 c0 a8 ....@... ..
00 02 0a 8e 06 fa 33 d8 dc b6 51 89 f5 75 50 18 ..3. ..Q...uP.
40 24 38 37 00 00 71 67 72 75 67 00 @$87..qg rug.

```

Figure 6. Frame with steganogram

5. Conclusion

As shown in [5] total Internet traffic contains about 7% retransmission data network caused by excessive delays or packet reordering. It is necessary to use RSTEG method taking into account this fact, so as not to arouse suspicion at a listening side, it is best to do 2-4 messages with steganogram on 100 legitimately transferred messages.

To complicate the task of listening side it is possible to send a key periodically, which will run dropping packets routine, thus causing repeated messages. That way we increase the cost of the analysis of retransmission packets that don't carry a secret message.

A variation of RSTEG steganalysis is a statistical analysis of all retransmission packets for this session and selection adjacent, repeatedly sent messages with different payload fields. But this tremendously reduces speed, especially if we

have to analyze traffic of a large hub.

Because the checksum of the retransmitted packet generated by RSTEG is different from that of the original packet, one can make the detection just by comparing the captured retransmitted checksum with that of the recoded one. So, it is possible to modify implementation of this method, so that except a payload field, the checksum is also changed [6].

References

- [1] W. Mazurczyk, M. Smolarczyk, K. Szczypiorski, "Retransmission steganography and its detection," Soft Computing, ISSN: 1432-7643 (print version), ISSN: 1433-7479 (electronic version), Journal no. 500, Springer, November 2009.
- [2] D. Wetherall, A. Tanenbaum, (2011). Computer networks. Upper Saddle River, NJ: Pearson Prentice Hall. ISBN 0-13-212695-8.
- [3] SharpPcap library, <http://sourceforge.net/projects/sharppcap/>
- [4] WinDivert library, <http://reqrypt.org/windivert.html>
- [5] C. Chen, M. Mangrulkar, N. Ramos and M. Sarkar. Trends in TCP/IP Retransmissions and Resets, Technical Report, <http://cseweb.ucsd.edu/classes/wi01/cse222/projects/reports/tcp-flags-13.pdf>
- [6] Jiangtao Zhai, Guangjie Liu, Yuewei Dai. An Improved Retransmission-based Network Steganography: Design and Detection. Journal of Networks, Vol. 8, No 1 (2013), 182-188, Jan 2013.