POSTER: An Educational Network Protocol for Covert Channel Analysis Using Patterns

Steffen Wendzel
Fraunhofer FKIE / Hochschule Worms, Germany
wendzel@hs-worms.de

Wojciech Mazurczyk
Warsaw University of Technology, Poland
wmazurcz@elka.pw.edu.pl

ABSTRACT

The utilization of information hiding is on the rise among cybercriminals, e.g. to cloak the communication of malicious software as well as by ordinary users for privacy-enhancing purposes. A recent trend is to use network traffic in form of covert channels to convey secrets. In result, security expert training is incomplete if these aspects are not covered. This paper fills this gap by providing a method for teaching covert channel analysis of network protocols. We define a sample protocol called Covert Channel Educational Analysis Protocol (CCEAP) that can be used in didactic environments. Compared to previous works we lower the barrier for understanding network covert channels by eliminating the requirement for students to understand several network protocols in advance and by focusing on so-called hiding patterns.

CCS Concepts

•Security and privacy → Network security; •Social and professional topics → Computing education;

Keywords

Covert Channels; Steganography; Information Hiding

1. INTRODUCTION

Network covert channels are communication paths that allow a hidden and unforeseen data exchange in computer networks. These channels are created by so-called hiding methods and belong to the research domain of Network Steganography [2, 1]. Covert channels can enable stealthy malware communications, constant unnoticeable data leakage from organizations, hidden communications of intelligence organizations, or covert communications for journalists to transfer illicit information under censorship [2]. Recently, we are witnessing a raising interest from the security community in information hiding techniques and a rising utilization of the methods by cybercriminals. However, when training students and security experts these aspects are covered very briefly or are not covered at all. That is why it is vital to devise new ways in which this training gap can be efficiently filled.

CCS'16 October 24-28, 2016, Vienna, Austria

ACM ISBN 978-1-4503-4139-4/16/10.

DOI: http://dx.doi.org/10.1145/2976749.2989037

In order to understand covert channels and their overall impact on security, an in-depth understanding of a number of topics related to network communications, including several network protocols, and fundamentals of steganography is required. This precondition is a challenge for teaching covert channels to students.

We present a new approach to teach network covert channels. Our solution eliminates the requirement of knowing and understanding several separate network protocols such as IP, TCP or HTTP in advance. Instead, we designed a single network protocol that is intentionally vulnerable to a number of covert channels at once. In addition, instead of teaching students the more than 130 known hiding methods, our approach focuses on several generic hiding *patterns*. Patterns represent the core ideas of hiding methods and reduce the number of concepts that students must comprehend. We demonstrate how our educational protocol can be used to generate especially covert *storage* channels, i.e. those channels that embed hidden data in a network protocol data unit (PDU) instead of modifying packet timings (*timing* channels). Our proof-of-concept implementation is available as an open source code and it is designed for teaching in undergraduate classes.

The reminder of this paper is structured as follows. Section 2 discusses related work while Section 3 introduces our educational network protocol CCEAP. Section 4 explains the link between CCEAP and hiding patterns. Section 5 shows the envisaged teaching workflow and sample exercises while Section 6 concludes.

2. RELATED WORK

Several suveys on network covert channels exist, e.g. [1, 2, 3]. Within these publications, the authors discovered more than hundred hiding methods. As shown in [3], 109 hiding methods could be broken down to eleven different *hiding patterns*. Such hiding patterns describe the core idea of a hiding method and are the actual aspect that must be understood by students. For instance, while several hiding methods modify the case of ASCII letters in plaintext protocols to signal hidden data, the core idea of case modification is always represented by the same pattern '*Value Modulation*'.

Few publications deal with the teaching of network covert channels. In 2008, Zander and Armitage published the so-called *Covert Channel Evaluation Framework* [4]. The framework can be used to establish and analyze a variety of network covert channels over TCP/IP. Recently, Zseby *et al.* were the first to provide an educational testbed for network steganography [5]. Their testbed is available for download and was evaluated with university students.

We do not consider our work as a replacement with the previous efforts by Zander *et al.* or Zseby *et al.* but rather as a complementary approach. It must be emphasized that we provide a different angle for teaching network covert channels with a single network protocol which is especially designed for educational purposes.

3. PROTOCOL DESIGN

Existing network protocols are vulnerable to several hiding patterns simultaneously only but few protocols are vulnerable to many patterns. For this reason, teaching hiding methods requires switching between several network protocols to describe the analysis process. Switching between different network protocols within the same tutorial or class is a challenge as students must process each of these protocols and have to be taught their basics a priori. This obstacle makes it harder to grasp the main concepts of information hiding. That is why we propose and present a protocol to illustrate the process of covert channel analysis. CCEAP, the *covert channel educational analysis protocol*, is designed in a way that it is vulnerable against several of the known hiding patterns described in [2, 3]. In comparison to previous approaches, we

- devise the CCEAP communication protocol that is designed for teaching covert channel analysis and that allows to understand several hiding methods with the use of a single protocol instead of focusing on TCP, IP, UDP and several other protocols;
- implemented CCEAP in a tool which is designed in a way to make it as accessible as possible by limiting the lines of code and delivering all relevant data directly as textual output;
- integrated no protocol elements in CCEAP which are not necessary for education (existing protocols do contain such elements);
- make hiding patterns the central aspect by exemplifying hiding methods in CCEAP with a link to a pattern;
- ease the understanding of hiding patterns in two scenarios: either, students need to find ways how to realize a specific hiding method with CCEAP, or, they are shown commands or traffic of CCEAP and then need to determine the related pattern (such as in case of a malware analysis in practice);
- allow the hybrid combination of hiding methods.

CCEAP comprises a simple, extendable protocol header (Figure 1). The main header contains three 32-bit words. Word 0 consists of a **sequence number** used to order packets at the receiver-side. The **number of options** field indicates whether one or more 'options' headers are present which are used to extend the main header. These options headers are embedded behind the main header, similarly to IPv4 options. The field **destination length** indicates the length of the destination address, which is an ASCII value of 1-8 bytes length. The **dummy** byte is intentionally reserved for future use like it is the case with several existing network protocols. Words 1 and 2 contain the **destination address** which is filled with padding bits if the address is less than eight bytes long.

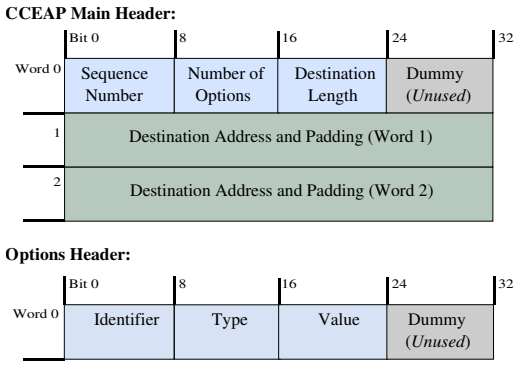

Figure 1. CCEAP main header (top) and options header (bottom)

A CCEAP packet can contain up to 255 options headers (their number is indicated by the **number of options** field in the main header). Each option contains a freely choosable **identifier**, a freely definable **type**, and a freely choosable **value**. To end up with a full word, the last byte is again an unused **dummy** value.

Table 1 shows how the particular fields of the CCEAP main and options headers reflect fields in existing network protocols.

Main header
Sequence Number: TCP and IPSec AH 'Sequence Number'
Number of Options: IPv4 'Internet Header Length' field
Destination Length: IPv6 'Payload Length', DNS' resource record domain name encoding (similar); DHCP option's padding
Dummy: Undefined/reserved bits in IPv6, TCP, IEEE 802.5, RIP and several other network protocols
Destination Addr./Padding: ASCII value in plaintext headers, e.g. HTTP, NNTP FTP, POP or SMTP
Options header
Identifier: IPv4 'Identifier', TCP ISN (similar)
Type: IPv4 'Option Number'
Value: IPv4 'Option Data'; ressource records in DNS
Dummy: same as in case of the main header

Table 1. Relation of CCEAP header fields to header fields in common network protocols

4. ADDRESSING HIDING PATTERNS

Hiding patterns are classified into those that establish network covert storage channels and those that establish network covert timing channels. Timing channels can either be dependent on a particular network protocol (*protocol-aware*) or they can be independent of a network protocol (*protocol-agnostic*) [2]. The implementation covers all patterns that either establish covert storage channels or protocol-aware covert timing channels and that are not specific for lower OSI layers (≤ 2). Table 2 explains the known hiding patterns and describes how CCEAP is vulnerable against each pattern.

Size Modulation (*Modulate size of a PDU or particular header element*): Modulation of address length or number of options
Sequence Modulation (*Alternation of header field order in a PDU*): Alternation of order in which n options are encapsulated
Add Redundancy (*Creation of space within a PDU/PDU field by addition of redundant bits*): Adding a new option of type X
Random Value (*Modification of a random value field*): Placing covert data in the initial sequence number
Value Modulation (*Selection of allowed PDU element values*): Change the case of ASCII letters used for the destination address
Reserved/Unused (*Utilization of reserved/unused bits in PDU fields*): Overwriting the reserved main header field 'dummy'
Artificial Message/Pkt Loss (*Artificial loss of network packets*): Messages are sent with incremental sequence values. The user can force the program to exclude a selected sequence number, indicating a message loss
Artificial Retransmission (*Introduction of artificial message duplications*): The user can select the sequence number of a packet to be duplicated and that will thus be sent twice
Manipulated Message Ordering (*Modulation of message sequences*): Define the order of sequence numbers with which messages are sent (instead of the default incremental values)

Table 2. How CCEAP addresses selected hiding patterns of [2].

5. WORKFLOW AND EXERCISES

We now explain the foreseen workflow for teaching CCEAP with a tool that implements the protocol and present sample exercises.

5.1 Teaching Workflow

We first describe the tasks that need to be performed by the lecturer, followed by the tasks that must be fulfilled by the students.

Lecturer: Initially, the lecturer has to teach the functioning, structure and syntax of CCEAP and of the hiding patterns. Due to the simplicity of CCEAP, this task can be achieved in little time. It is up to the lecturer to decide how CCEAP is taught. Teaching could be done in form of a lecture, via handout or by providing the students with a link to the protocol description. Next, the lecturer assigns a pattern-related exercise to the students (cf. Section 5.2). These exercises are structured in the form of a sentence, e.g.: '*Find a way to establish a covert channel that uses the pattern X using CCEAP. Verify your approach with the CCEAP tool.*' Alternatively, the exercise can be *reversed* by showing a sample execution of the tool and let the student determine the represented pattern. After a student submitted a solution, the lecturer verifies the answer and can – if necessary – guide the student to a better or correct answer.

Students: The students, on the other hand, study CCEAP and the command-line options of the CCEAP tool. The tool's client sends the data to its server that evaluates the received message and displays the received content. The tool's command line options allow to highly influence the header structure and field values of CCEAP. After the students received the exercise, they try to find a way to create a covert channel with the tool that represents the pattern selected by the teacher. If the exercise was reversed, the students analyze the behavior of CCEAP using the tool and match it to the description of the hiding patterns. This step can be repeated to address different hiding methods. The students submit their solutions to the lecturer to receive feedback (and help).

5.2 Sample Exercises

The goal of exercises is to learn how a covert channel can be created and to understand the idea behind a pattern to such an extent that students can apply it. In a reversed scenario, the students must be able to tell which hiding pattern was used for a given covert channel. While the first scenario reflects the covert channel analysis of protocols, the reversed scenario reflects scenarios like malware/data leakage analysis. In real-world scenarios, students would similarly analyze the behavior of data packets sent by a malware and then determine what type of covert channel was applied. However, the fundamental aspects would not differ to the CCEAP approach from the exercise. The following four exercises underpin the concept of both, the standard and the reversed scenario.

Ex. 1: Value Modulation. The students are asked to create a covert channel that represents a Value Modulation. First, the students verify the definition of the pattern as taught or via available descriptions from a handout, website (http://ih-patterns.blogspot.com/) or papers [2, 3]. Second, they determine how CCEAP can be used to represent the pattern. A typical scenario for a Value Modulation mentioned in publications is the alternation between upper and lower case characters in protocols headers. CCEAP contains the 'Destination Address' field and the students decide to send data either to address 'ABC' (or 'abc') to signal a covert '0' (or '1') bit. Third, to test their idea, the students check the parameters provided by the CCEAP tool and detect the parameter `-d` that specifies the destination. The students start the server and let the client send two packets to the server, once with the former and once with the latter destination address: `$ client -D 127.0.0.1 -P 3333 -d ABC ; client -D 127.0.0.1 -P 3333 -d abc`, where `-D` and `-P` indicate the address and port of the server to be used. Both, `-D` and `-P` are left out in the reminder to increase readability. On the server, the students can check the output that displays the received packets and submit the answer.

Ex. 2: Sequence Modulation. The students are asked to use a Sequence Modulation pattern. They analyze CCEAP to determine whether a header element represents a definable sequence of elements. The analysis reveals that the order of options can be freely defined. The students define the option types 6 and 7 and signal a '0' bit if the order is 6, 7 and a '1' bit if the order is 7, 6. They use the 'identifier' field in an incremental way, starting with 0 and set the 'value' field to 0 as both fields are not of relevance for the exercise. To signal a '0' bit followed by a '1' bit, they call the client as follows: `client -o 0,6,0/1,7,0 ; client -o 0,7,0/1,6,0`.

Ex. 3: Reverse Scenario. This exercise lists a command that was used to create a covert channel: `client -i 10 -p 11 -c 20`. The students are asked to name the represented hiding pattern and – if possible – conclude what hidden bits were transferred. The students find out that this command sets the initial sequence number to 10, duplicates the packet with the sequence number 11 and transfers 20 packets. By default the client uses an incremental message order and transfers 10 packets instead of 20. Searching for a matching pattern, the students find the Artificial Retransmission pattern in which packets are duplicated. So, the hidden code that was transferred is 20 bits of which one bit was sent twice, resulting in the message '0100...0000' or '1011...1111'.

Ex. 4: Hybrid Patterns. In an extended scenario, an exercise can feature hybrid patterns in which the students have to combine multiple hiding patterns simultaneously (e.g. by overwriting the 'dummy' field (Reserved/Unused pattern) and at the same time manipulating the number of options in the packet to influence the PDU size (Size Modulation pattern). The correct combination of command line parameters for sending a packet with dummy value 1 or 2 and transferring either zero or one options would be: `client -u 1` and `client -u 2 -o 1,2,3`, and any possible variation of the two commands and their two parameters.

6. CONCLUSION AND OUTLOOK

CCEAP (http://ih-patterns.blogspot.com/) can be used to teach pattern-based covert channel analysis with a single network protocol instead of several protocols. We currently perform an evaluation phase with the tool at two universities. Future work comprises to develop an educational approach for teaching low-level and protocol-agnostic covert channels in a single tool.

7. REFERENCES

[1] B. Carrara and C. Adams. A survey and taxonomy aimed at the detection and measurement of covert channels. In *Proc. IH&MMSec'16*, pages 115–126. ACM, 2016.

[2] W. Mazurczyk, S. Wendzel, S. Zander, et al. *Information Hiding in Communication Networks: Fundamentals, Mechanisms, and Applications*. Wiley-IEEE, 2016.

[3] S. Wendzel, S. Zander, B. Fechner, and C. Herdin. Pattern-based survey and categorization of network covert channels. *ACM Computing Surveys*, 47(3):50:1–50:26, 2015.

[4] S. Zander and G. Armitage. CCHEF – covert channels evaluation framework: Design and implementation. Technical Report TR 080530A, 2008.

[5] T. Zseby, F. I. Vazquez, V. Bernhardt, et al. A network steganography lab on detecting TCP/IP covert channels. *IEEE Transactions on Education*, 2016.