Covert Channel

Pallavi Arivukkarasu
Department of EECS
Washington State University
Pullman, WA, USA
p.arivukkarasu@wsu.edu

***Abstract*—This paper describes about the design and implementation of a covert channel. Covert channels can be used to break the "no writing down" confidentiality. Covert channels can also help break containment and violate isolation on cloud computing platforms. The confinement problem is a relevant cyber security problem.**

Covert channels use a shared resource that is not intended to communicate information in an unintended way. Covert channels are typically timing-based or storage-based and can be noisy or noiseless.

Cyber security is an "arms race," with attackers developing new malware that attempts to avoid detection by cyber defenders. When detected, attackers must develop new TTP (i.e., tools, tactics, and procedures) to operate once again undetected. Attack and defense TTP are coevolutionary.

***Index Terms*—covert channel, TCP, confidentiality, cybersecurity**

I. Introduction

Covert communication is the exchange of information/data using a covert channel. A covert channel is a type of computer attack/threat that enables the communication between various objects and processes that were not allowed to communicate because of the system/network's policy. There are two types of covert channels - storage and timing channels. [2]

The type of covert channel presented here is a timing channel that makes use of TCP.

Craig Rowland's [3] implementation of the covert channel utilizes the IP layer's Identification field, which is a 16-bit field capable of holding 2 ASCII characters per packet. Rowland also mentioned in his documentation that the TCP initial sequence number field and TCP Acknowledge number field are viable options for hiding data for a covert channel. The problem he mentioned is that the packets sent are not reliable because he does not ACK the packets for retries. The covert channel implemented here uses the source port field while enabling the "E" (ECN / Explicit Congestion Notification) flag to denote that the packet is a covert message. [4]

II. Implementation

The program was written in python with the help of scapy, a packet crafting API. There are two programs:

client.py – the covert message sender

server.py – the covert message receiver.

The client program crafts a packet so that the "E" flag (ECN) denotes that the packet is a covert message and the message itself is stored in the source port. The E flag bit is the choice of flagging a covert message for two fundamental reasons:

1. It is rarely used in standard TCP/IP connections.

2. It requires strict compliance on ECN processing between two communicating machines.

The ports are redundant for this implementation since we are not establishing a three-way handshake; instead, we expect an RST and ACK back from the server. The message itself was implemented in the source port and will not alter the program's behavior if it was implemented in the destination port. The destination port in this implementation is entirely random.

III. Execution

Scapy must be pre-installed prior to running the program. Scapy can be found at https://scapy.net/ [1].

Unzip the zip file and run: python setup.py install The programs can be executed using the following commands.

$ python client.py <host_ip>

$ python server.py

Note: the user must be a root user to use the program.

The easiest way to identify that the program itself is to input a pattern ("abc" and "123"), such that the values of the characters will be sequential.

IV. Conclusion

Like Rowland's implementation, this implementation suffers heavily on reliability and negates the bounce feature presented in Rowland's implementation. Future changes of this program should feature a bounce implementation and reliability. Thus, the design and implementation of a covert channel has been completed.

References

[1] https://scapy.net/

[2] https://study.com/academy/lesson/covert-communication-channels-definition-issues-tunneling.html

[3] Rowland, Craig H. "Covert channels in the TCP/IP protocol suite." (1997).

[4] Tumoian, Eugene, and Maxim Anikeev. "Network based detection of passive covert channels in TCP/IP." In The IEEE Conference on Local Computer Networks 30th Anniversary (LCN'05) l, pp. 802-809. IEEE, 2005.