COMP 8505 ASSIGNMENT ONE TESTING DOC

Yiaoping Shu

A00930347 September 22 2018

Introduction

In Rowland's paper, he describes the weaknesses of the TCP/IP protocol suite which allows an attacker to perform techniques of covert channels by storing sensitive data in packets unknown to others. A covert channel is any type of channel communicating with one another to transfer information that violates the security policy of the system. Rowland says that covert channels can be implemented by hiding data inside the TCP headers. Further detail about how the covert channel is implemented will be shown below.

The covert channel program is written in Python, with a client side and a server side. The client-side readers in data input from the command terminal and manipulate the packet by storing the data inside the source port field. The packet being forged also has a flag added to it. We use the TCP flag 0x80 to allow the receiver to know that this is a covert message being sent.

The server side of the program is constantly sniffing for packets and if the packet being sniffed is TCP and has the flag indicating that it's a covert message, it will print the message to terminal for user to read.

To allow for a more secure way of ensuring the attackers don't get caught, a simple implementation of having the messages being sent at a random time is performed.

Packet manipulation in this program is done by using an API called Scapy, which allows us to manipulate the packets in python.

Usage

Before running the program, ensure you are the root user and have the latest Scapy program installed. Unzip the Scapy file and navigate to the folder. Type in the following

#pip3 install scapy-python3

To run the program, unzip the file containing the source code and navigate to it in command line. Type in the following commands after you are in the proper folder to run the client:

#python3 Client.py [*destination IP address*] [Random timer one] [Random timer two]

IP Address is the IP address that you are trying to send the message to. Random timer one and two is the range of random numbers that you want the packets being sent to be randomized. For example, if 3 and 5 are the two numbers typed, each packet will be sent at a random second between 3 and 5.

To run the server, type in the following command:

#python3 Server.py

Testing

Test #	Description	Expected Result	Result (Pass/Fail)
1	Navigate to source folder. Execute client program by typing in “python3 client.py” with proper arguments	User can run the client program	Pass
2	Navigate to source folder. Execute server program by typing in “python3 server.py”	User can run the server	Pass
3	Run the client program with any incorrect arguments	Inputting incorrect client-side arguments results in error (usage thrown)	Pass
4	Run the client program and send message. Ensure target machine is running.	User can send packets from client	Pass
5	Run client program as well as target machine. Check client side to ensure entire message is sent with notice displayed to user	Entire message can send to target machine	Pass
6	Run server and client program. Send message from client and ensure server can detect and decode packets.	Server side can sniff packets	Pass
7	Wireshark Captures	Check below for Wireshark captures and explanations	

Screenshots

Test #1

```
[root@localhost a1]# python3 client.py 192.168.0.1 0 4
WARNING: No route found for IPv6 destination :: (no default route?). This affect
s only IPv6
Enter your message:this is a test
```

Test #2

```
WARNING: No route found for IPv6 destination :: (no default route?). This affect
s only IPv6
```

Test #3

```
[root@localhost a1]# python3 client.py 192.168.0.1
WARNING: No route found for IPv6 destination :: (no default route?). This affect
s only IPv6
Usage: [host_ip] [RandTimerStartRange] [RandTimerEndRange]
```

Test #4

```
[root@localhost a1]# python3 client.py 192.168.0.1 0 1
WARNING: No route found for IPv6 destination :: (no default route?). This affect
s only IPv6
Enter your message:this is a test sending
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets
```

Test #5

```
Sent 1 packets.

Packet finished sending

Enter your message:testing
```

Test #6

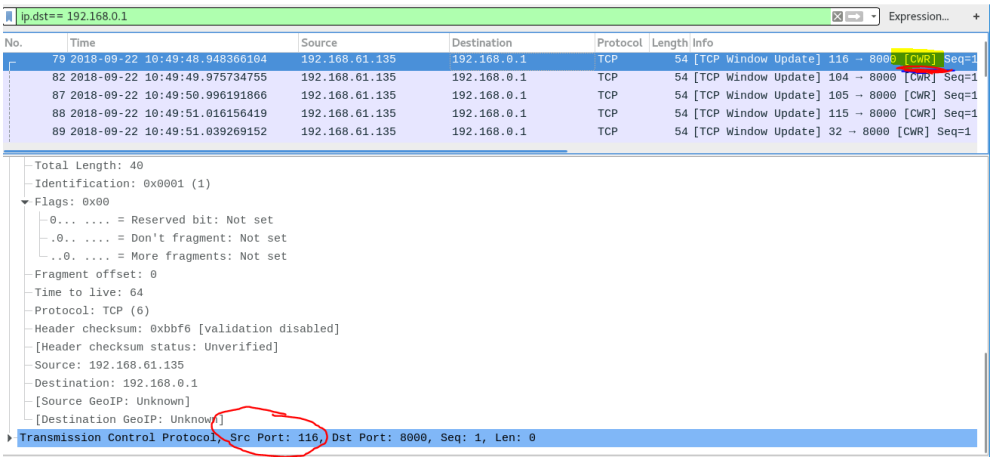
```
[root@localhost a1]# python3 server.py
WARNING: No route found for IPv6 destination :: (no default route?). This affect
s only IPv6


this is a test sending
```

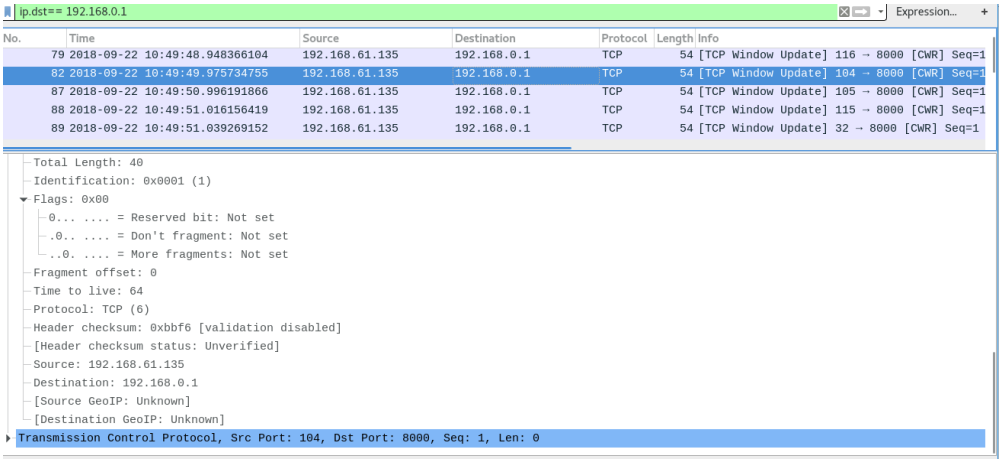
Wireshark Captures:

Our message being sent: “this is a test”

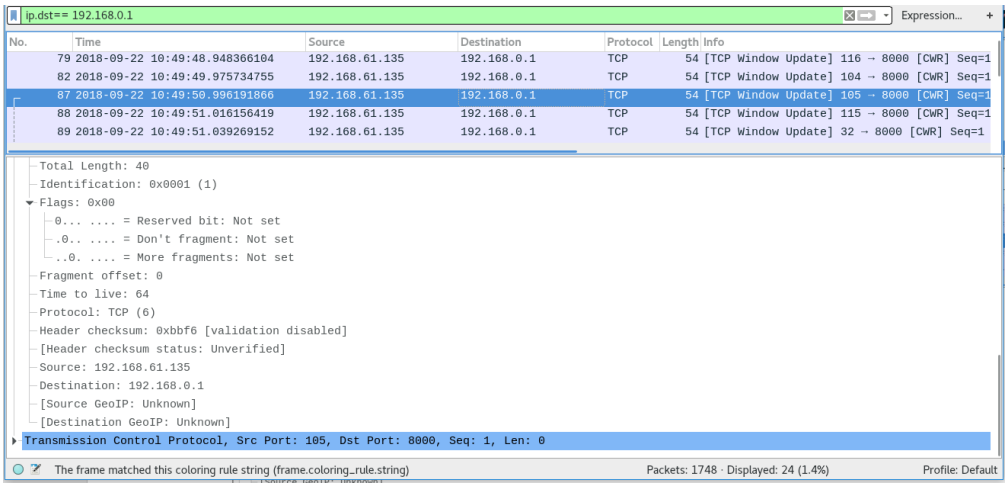
The below shows the message that was being sent from the above tests. The first word that we sent was “this” and is stored in the source port with their Ascii code representation. Also note the flag that denotes our message is of covert channel, using a flag of 0x80 = CWR.



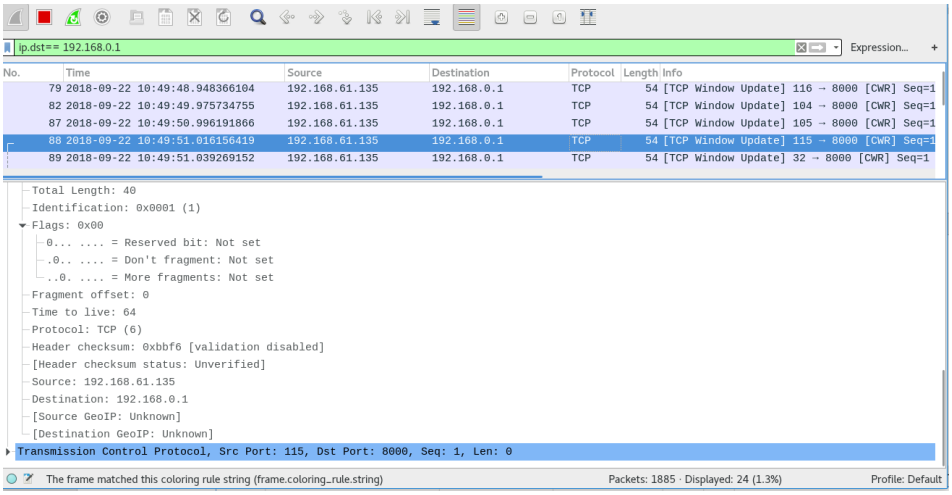
Source Port 116: The Ascii code of letter t is 116



Source Port 104: The Ascii code of letter h is 104



The ascii code of letter i is 105



The Ascii code of letter s is 115.