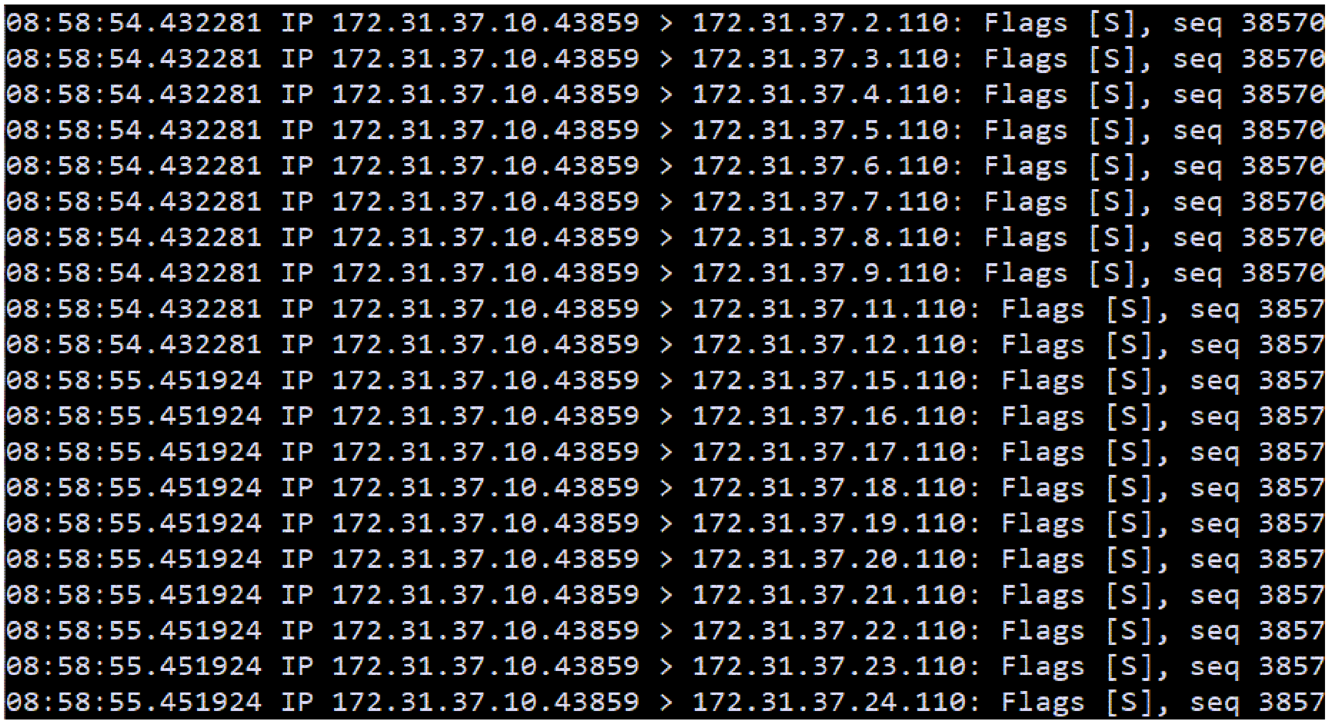
Exploring TCP Flags in Malicious Traffic

For this challenge you will remain on the Ubuntu endpoint.

This next packet capture contains data from a compromised host that was used to scan the internal network for available endpoints and open ports.

1. The first sign of anomalous activity can be seen by taking a general look at the capture and focusing on the TCP flags:

   `tcpdump -nn -r syn-scan.pcap | more`

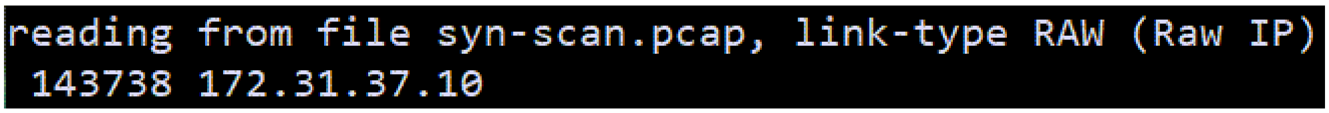


```
08:58:54.432281 IP 172.31.37.10.43859 > 172.31.37.2.110: Flags [S], seq 38570
08:58:54.432281 IP 172.31.37.10.43859 > 172.31.37.3.110: Flags [S], seq 38570
08:58:54.432281 IP 172.31.37.10.43859 > 172.31.37.4.110: Flags [S], seq 38570
08:58:54.432281 IP 172.31.37.10.43859 > 172.31.37.5.110: Flags [S], seq 38570
08:58:54.432281 IP 172.31.37.10.43859 > 172.31.37.6.110: Flags [S], seq 38570
08:58:54.432281 IP 172.31.37.10.43859 > 172.31.37.7.110: Flags [S], seq 38570
08:58:54.432281 IP 172.31.37.10.43859 > 172.31.37.8.110: Flags [S], seq 38570
08:58:54.432281 IP 172.31.37.10.43859 > 172.31.37.9.110: Flags [S], seq 38570
08:58:54.432281 IP 172.31.37.10.43859 > 172.31.37.11.110: Flags [S], seq 3857
08:58:54.432281 IP 172.31.37.10.43859 > 172.31.37.12.110: Flags [S], seq 3857
08:58:55.451924 IP 172.31.37.10.43859 > 172.31.37.15.110: Flags [S], seq 3857
08:58:55.451924 IP 172.31.37.10.43859 > 172.31.37.16.110: Flags [S], seq 3857
08:58:55.451924 IP 172.31.37.10.43859 > 172.31.37.17.110: Flags [S], seq 3857
08:58:55.451924 IP 172.31.37.10.43859 > 172.31.37.18.110: Flags [S], seq 3857
08:58:55.451924 IP 172.31.37.10.43859 > 172.31.37.19.110: Flags [S], seq 3857
08:58:55.451924 IP 172.31.37.10.43859 > 172.31.37.20.110: Flags [S], seq 3857
08:58:55.451924 IP 172.31.37.10.43859 > 172.31.37.21.110: Flags [S], seq 3857
08:58:55.451924 IP 172.31.37.10.43859 > 172.31.37.22.110: Flags [S], seq 3857
08:58:55.451924 IP 172.31.37.10.43859 > 172.31.37.23.110: Flags [S], seq 3857
08:58:55.451924 IP 172.31.37.10.43859 > 172.31.37.24.110: Flags [S], seq 3857
```

   Analysis: The continuous requests from a single IP with the SYN flag set to multiple IPs and ports without getting any successful SYN/ACK packets back is a clear indicator of something called SYN scanning. Using this method, an adversary will send packets with the SYN flag set and wait to see if there are any responses. However, with any successful responses the adversary will not complete the three-way handshake since the information has already been gained letting them know the port is open.

2. How could you validate the IP performing the scanning?

   `tcpdump -nn -r syn-scan.pcap 'tcp[tcpflags] == tcp-syn' | cut -f 3 -d " " | cut -f 1-4 -d "." | sort | uniq -c | sort -n`



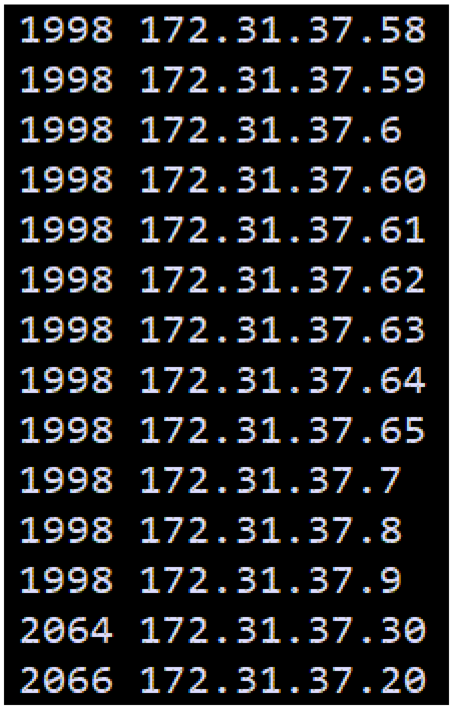
```
reading from file syn-scan.pcap, link-type RAW (Raw IP)
 143738 172.31.37.10
```

Analysis: 143,738 of the SYN packets sent in this PCAP all belong to a single host: 172.31.37.10

3. Can you return a list of the scanned IPs?

   Using the same method as before, but change the first column cut to be the destination IPs:

   ```
   tcpdump -nn -r syn-scan.pcap 'tcp[tcpflags] == tcp-syn' | cut -f 5 -d " " |
   cut -f 1-4 -d "." | sort | uniq -c | sort -n
   ```

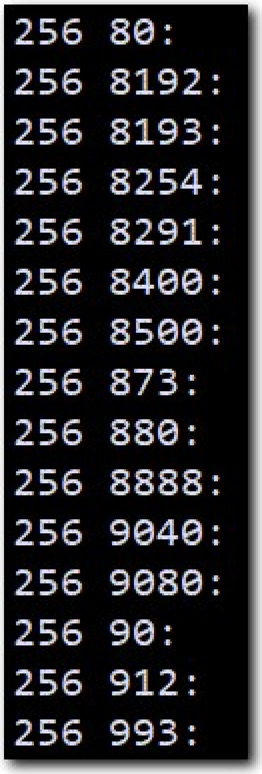
   

   ```
   1998 172.31.37.58
   1998 172.31.37.59
   1998 172.31.37.6
   1998 172.31.37.60
   1998 172.31.37.61
   1998 172.31.37.62
   1998 172.31.37.63
   1998 172.31.37.64
   1998 172.31.37.65
   1998 172.31.37.7
   1998 172.31.37.8
   1998 172.31.37.9
   2064 172.31.37.30
   2066 172.31.37.20
   ```

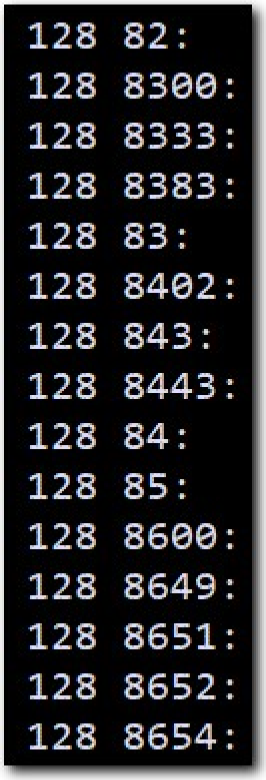
   Analysis: Having 1,998 connections to each of the destination IPs is an indicator of scripted activity. Simply seeing the count is higher for .30 and .20 lets you know additional conversations happened there.

4. Can you return a list of the scanned ports?

   ```
   tcpdump -nn -r syn-scan.pcap 'tcp[tcpflags] == tcp-syn' | cut -f 5 -d " " |
   cut -f 5 -d "." | sort | uniq -c | sort -n
   ```



```
256 80:
256 8192:
256 8193:
256 8254:
256 8291:
256 8400:
256 8500:
256 873:
256 880:
256 8888:
256 9040:
256 9080:
256 90:
256 912:
256 993:
```

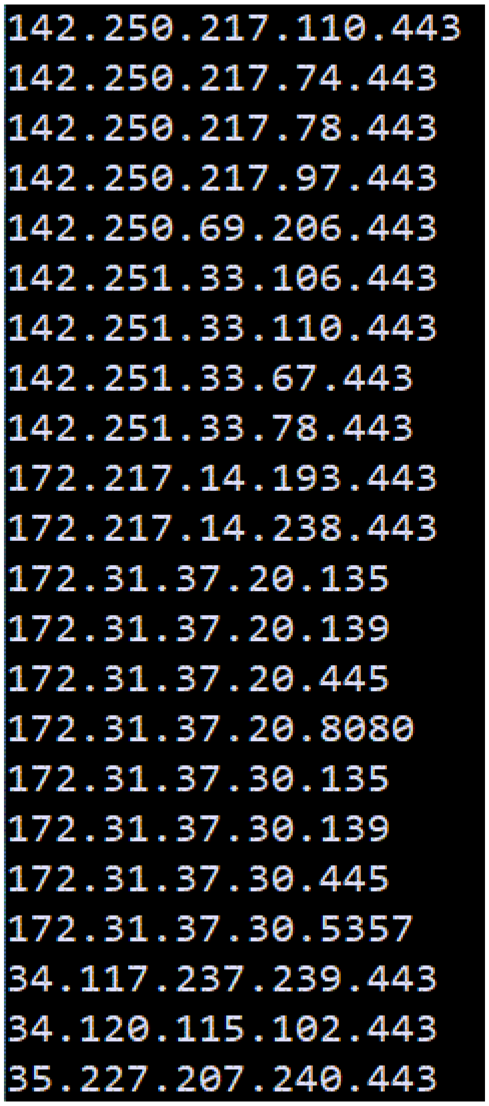


```
128 82:
128 8300:
128 8333:
128 8383:
128 83:
128 8402:
128 843:
128 8443:
128 84:
128 85:
128 8600:
128 8649:
128 8651:
128 8652:
128 8654:
```

Analysis: You can see scrolling through the returned list, multiple ports were attempted 256 times, and another set of ports was attempted 128 times. This seemingly scripted behavior is another clear indicator something strange is taking place.

5. Do you remember how to return a list of IPs that responded back and what ports are open?

```
tcpdump -nn -r syn-scan.pcap 'tcp[tcpflags] == (tcp-syn|tcp-ack)' | cut -f 3 -d " " | sort | uniq
```



```
142.250.217.110.443
142.250.217.74.443
142.250.217.78.443
142.250.217.97.443
142.250.69.206.443
142.251.33.106.443
142.251.33.110.443
142.251.33.67.443
142.251.33.78.443
172.217.14.193.443
172.217.14.238.443
172.31.37.20.135
172.31.37.20.139
172.31.37.20.445
172.31.37.20.8080
172.31.37.30.135
172.31.37.30.139
172.31.37.30.445
172.31.37.30.5357
34.117.237.239.443
34.120.115.102.443
35.227.207.240.443
```

Analysis: All of this activity should absolutely be investigated, however there are some important notes you should take away when considering this output.

- Most of the IPs respond back with port 443 (HTTPS) open. Since most of these are external IPs, it wouldn't normally seem malicious, but important to consider the timeline when investigating a compromised host and other activity.
- Focusing on the two internal IPs in the same network (172.31.37.20 and 172.31.37.30), both are responding on ports 135, 139, and 445. Port 8080 and 5357 also report back successfully.
- Analysis of this incident will change depending on classification of each host being a standard endpoint or a server hosting applications. However, in both instances, this list of ports being open and responding back is extremely strange.

Understanding the functionality of TCP and UDP as transport protocols is fundamental to identifying anomalies in network traffic.