COMP232 Individual coursework

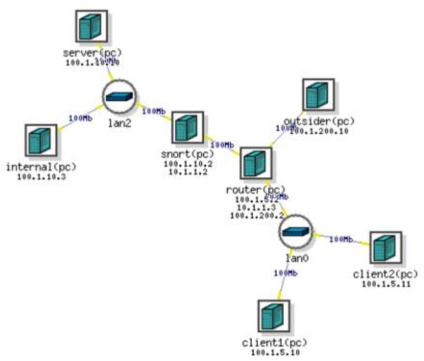
Assignment 3

Name: Yuyang.Wan

Student ID:20148429

1. Start Snort Without Rules

Step 1 Finish set up of new experiment



Step 2 Activate remote server Linux server and open an SSH client on your computer to connect to users.deterlab.net. Start snort.

```
[uolcomcq@users ~]$ ssh snort.idk.COMP232.isi.deterlab.net
uolcomcq@snort:~$
```

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

05/11-19:55:38.729597 100.1.5.11:38518 -> 100.1.10.10:7777
TCP TTL:62 TOS:0x0 ID:10375 IpLen:20 DgmLen:52 DF
***A**** Seq: 0xA631BE21  Ack: 0x55A5A5E7  Win: 0x1F5  TcpLen: 32
TCP Options (3) => NOP NOP TS: 2326679628 3205077543
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

05/11-19:55:38.729614 100.1.10.10:7777 -> 100.1.5.11:38514
TCP TTL:63 TOS:0x0 ID:32340 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xD76E4482  Ack: 0x338C0A59  Win: 0x1FB  TcpLen: 32
TCP Options (3) => NOP NOP TS: 3205077544 2326679430
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

05/11-19:55:38.729624 100.1.10.10:7777 -> 100.1.5.11:38518
TCP TTL:63 TOS:0x0 ID:23826 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x55A5AB8F  Ack: 0xA631BE21  Win: 0x1FA  TcpLen: 32
TCP Options (3) => NOP NOP TS: 3205077544 2326679469
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

05/11-19:55:38.729849 100.1.10.10:7777 -> 100.1.5.11:38514
TCP TTL:63 TOS:0x0 ID:32341 IpLen:20 DgmLen:1333 DF
***AP**F Seq: 0xD76E4A2A  Ack: 0x338C0A59  Win: 0x1FB  TcpLen: 32
```

Step 3 Run tcpdump to capture the data

```
1831 packets captured
1843 packets received by filter
0 packets dropped by kernel
uolcomcq@snort:~$ sudo /share/education/SecuringLegacySystems_JHU/process.pl /t
mp/dump.pcap
reading from file /tmp/dump.pcap, link-type EN10MB (Ethernet)
1589253996.728723 0
1589253997.760375 94
1589253999.792948 18
1589254000.793851 103
1589254002.826187 91
1589254003.846840 82
1589254004.942710 91
1589254005.944232 14
1589254006.951035 87
1589254008.983365 74
1589254010.028257 96
1589254011.992629 8
1589254013.010813 10
1589254015.094579 11
1589254016.095983 113
1589254018.112080 75
1589254019.145972 12
1589254021.111840 8
1589254022.177973 10
1589254023.438622 55
1589254024.438896 124
1589254025.443561 98
1589254027.290506 13
```

Full data

1589253996.728723 0
1589253997.760375 94
1589253999.792948 18
1589254000.793851 103
1589254002.826187 91
1589254003.846840 82
1589254004.942710 91
1589254005.944232 14
1589254006.951035 87
1589254008.983365 74
1589254010.028257 96
1589254011.992629 8
1589254013.010813 10
1589254015.094579 11
1589254016.095983 113
1589254018.112080 75
1589254019.145972 12
1589254021.111840 8
1589254022.177973 10
1589254023.438622 55
1589254024.438896 124
1589254025.443561 98
1589254027.290506 13
1589254028.292415 14

1589254030.308750 6
1589254031.363143 99
1589254033.392982 91
1589254034.395913 99
1589254036.454933 5
1589254037.465602 14
1589254038.481757 6
1589254039.525910 2
1589254040.527091 113
1589254042.560141 74
1589254043.574817 15
1589254043.617514 6

Questions:

1) What happens to the traffic to client1 when Snort is not running?

   Traffic to client1 rises rapidly when snort is not running.

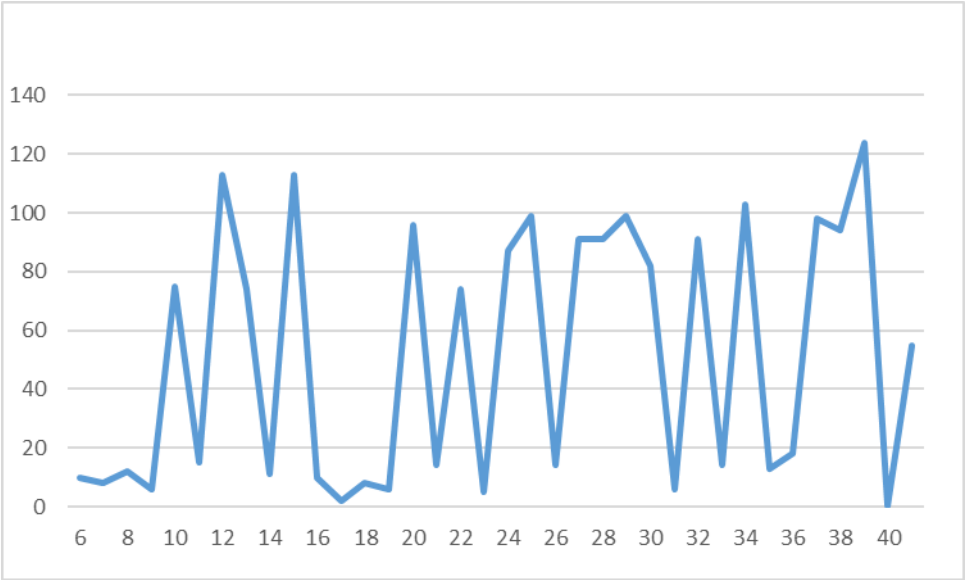
2) Is this a good thing?

   No, it can block the access to client1. Technically this can be developed into dos attack.

3) Based on Snort's output what can you say about the application? What port does it connect to?

   Snort can detect abnormal traffic to the client and restraints it. It is connected to the incoming port.

4) Please attach a graph of the traffic over time to your answers



5) What does the "-Q" option do in Snort?

-Q stand for Enable inline mode operation. In inline mode Snort creates a bridge between two network segments, and is responsible for passing traffic between the segments. It can inspect the traffic it passes, as well as drop suspicious traffic.

6) What does the "--daq nfq" option do in Snort?

--daq nfq means inline on Linux using netfilter. It provides three main functionalities including Packet filtering to Accepts or drops packets, NAT to Changes the source or destination IP address of network packets, Packet Mangling to Modifies packets for Quality of Service.

2. Analyze Network Traffic

Step1 Connect to router

```
Warning: Permanently added 'router.idk.comp232.isi.deterlab.net' (RSA) to the l
ist of known hosts.
```

Step2 Capture the data

```
ist of known hosts.
uolcomcq@router:~$ ifconfig
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.1.1.3  netmask 255.255.255.0  broadcast 10.1.1.255
        inet6 fe80::204:23ff:feae:cbf4  prefixlen 64  scopeid 0x20<link>
        ether 00:04:23:ae:cb:f4  txqueuelen 1000  (Ethernet)
        RX packets 142405  bytes 63236988 (63.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 145576  bytes 24509809 (24.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 100.1.200.2  netmask 255.255.255.0  broadcast 100.1.200.255
        inet6 fe80::204:23ff:feae:cbf5  prefixlen 64  scopeid 0x20<link>
        ether 00:04:23:ae:cb:f5  txqueuelen 1000  (Ethernet)
        RX packets 72713  bytes 12244819 (12.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 71129  bytes 32557463 (32.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.133  netmask 255.255.252.0  broadcast 192.168.3.255
        inet6 fe80::211:43ff:fed6:d635  prefixlen 64  scopeid 0x20<link>
        ether 00:11:43:d6:d6:35  txqueuelen 1000  (Ethernet)
        RX packets 24228  bytes 34687123 (34.6 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2399  bytes 211680 (211.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 100.1.5.2  netmask 255.255.255.0  broadcast 100.1.5.255
        inet6 fe80::211:43ff:fed6:d636  prefixlen 64  scopeid 0x20<link>
        ether 00:11:43:d6:d6:36  txqueuelen 1000  (Ethernet)
```

Questions:

7) The request that the client sends the server is broken into four parts. What are these parts and what order does they appear in? How are these parts seperated in the request?
8) Is this is a secure way for the client to send requests to the server? Explain your answer.

   I think it is a save way for client to send requests to server.

9) Can you recover one of the files sent by the server to a client? If so attach the file, a pcap the relevant packets and indicate which client this was sent to.