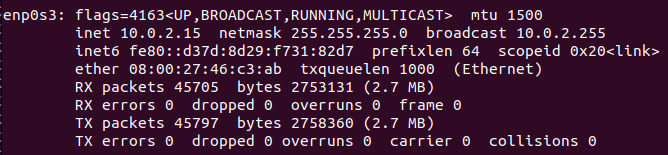
Snort IDS

The config file snort.conf



Ran ip route to find default gateway



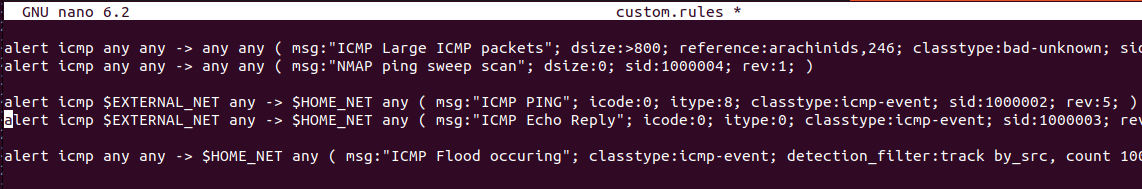
Ran ifconfig to find my ipv4 and subnet. Also note that I am using enp0s3



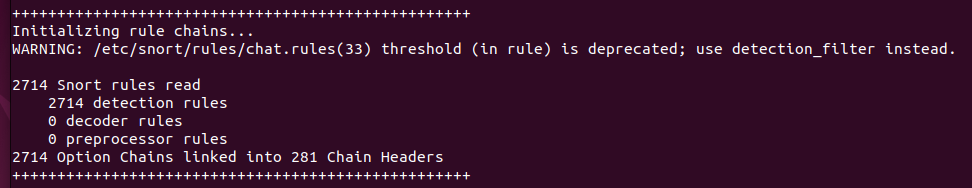
Set $HOME\_NET to the default gateway



Created my own .rules file



Here is a snapshot of my custom.rules file

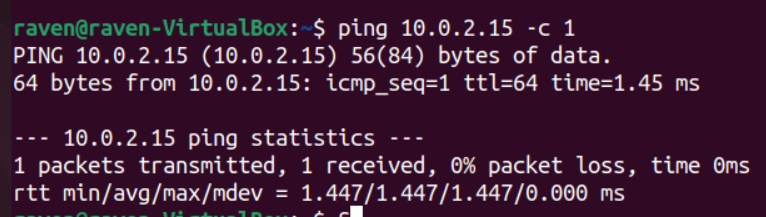


After writing and saving my rules, I run “snort -T -c /etc/snort/snort.conf -i enp0s3” to test. Other rules are also enabled, but we will not be testing these.

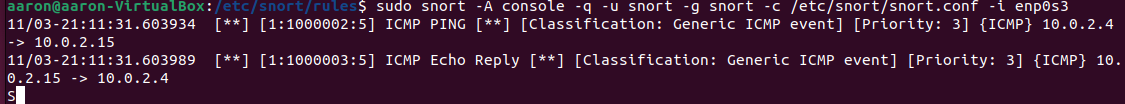


Run “snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i enp0s3” to run snort in alert mode with output to console.

Simple Ping Test



On my attacker, I pinged the victim’s ip address. (Ubuntu linux uses icmp for ping.)

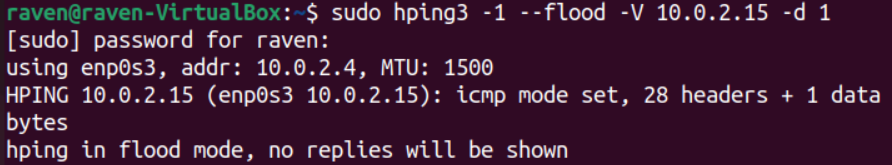


Snort successfully picked up the ping and echo based on the rules I set up. (Note that sid > 1000000, which means these are my custom rules and not existing Snort rules.)

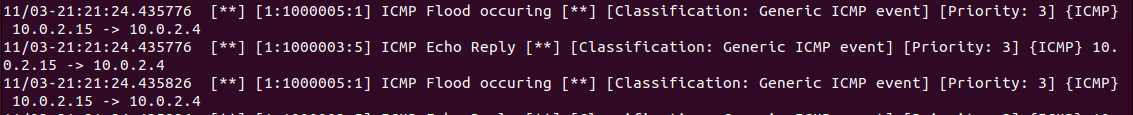
The following were the two rules used in the ping test:

* alert icmp $EXTERNAL\_NET any -> $HOME\_NET any (msg:"ICMP PING"; icode:0; itype:8; classtype:icmp-event; sid:1000002; rev:5;)
* alert icmp $EXTERNAL\_NET any -> $HOME\_NET any (msg:"ICMP Echo Reply"; icode:0; itype:0; classtype:icmp-event; sid:1000003; rev:5;)

ICMP-Based Attack



We hping3 to attempt a ICMP Flood attack on our victim. Note “-1” sets hping3 to icmp mode instead tcp by default. Also, “-d 1” to send packets of size 1.

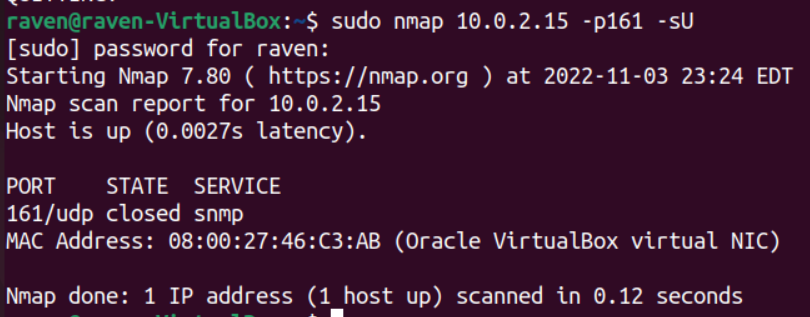
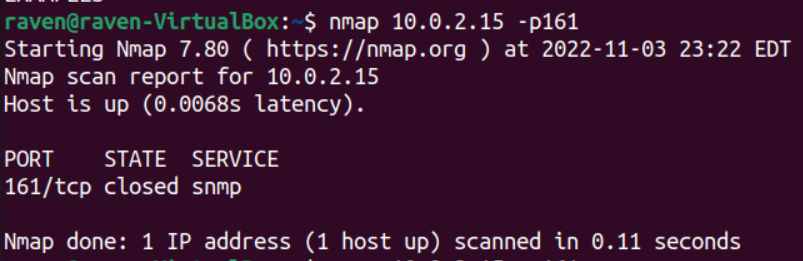


Snort successfully detected the ICMP Flood attack and the ip of the attacker.

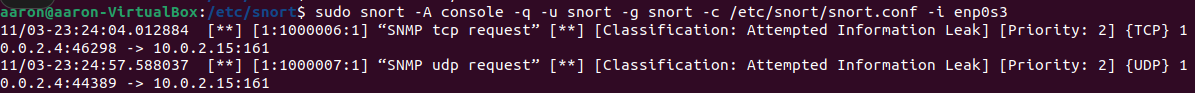
The following were the rules used in ICMP attacks

* # Alert when 100+ ICMP messages are sent under 3 seconds
  + alert icmp any any -> $HOME\_NET any ( msg:”ICMP Flood occuring”; classtype:icmp-event; detection\_filter: track by\_src, count 100, seconds 3; sid:1000005; rev:1;)

SNMP-Based Attack



I used nmap on the victim’s ip and port 161 which is the port that SNMP service uses. I tested with both tcp and udp versions of nmap scan.

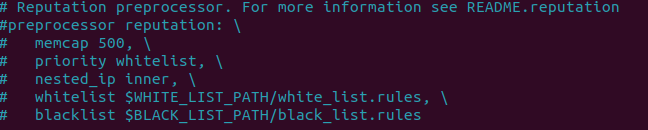


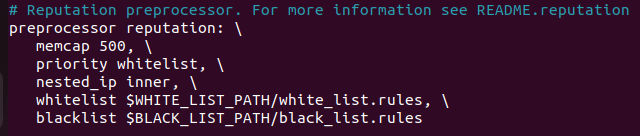
Snort successfully detected both versions of the nmap scan and identified the attacker.

The following were the rules used in ICMP attacks

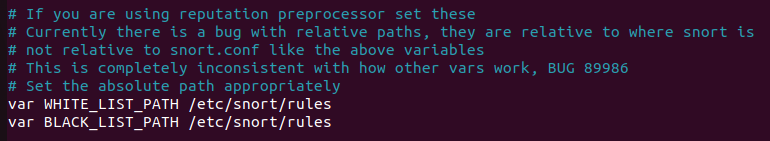
* alert tcp $EXTERNAL\_NET any -> $HOME\_NET 161 ( msg:“SNMP tcp request”; flow:stateless; classtype:attempted-recon; sid:1000006; rev:1 )
* alert udp $EXTERNAL\_NET any -> $HOME\_NET 161 ( msg:“SNMP udp request”; flow:stateless; classtype:attempted-recon; sid:1000007; rev:1 )

Attack from Blacklisted Source

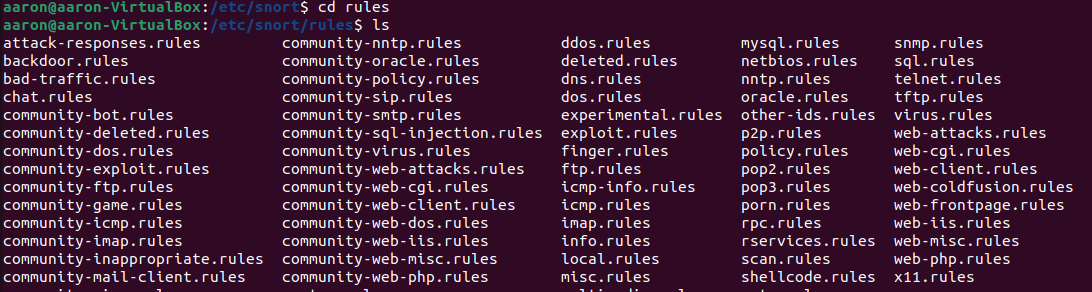




In the snort.conf file, the reputation preprocessor (which is used to select IP addresses to whitelist and blacklist) was initially commented out. I had to go uncomment it to enable its functionality.



Both whitelist & blacklist are stored in the rules directory.

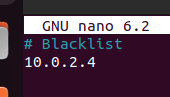


However, there are no black\_list.rules or white\_list.rules in the rules directory.

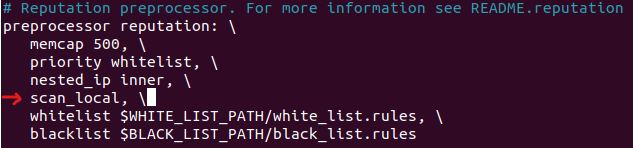


We make our own blacklist and whitelist files.

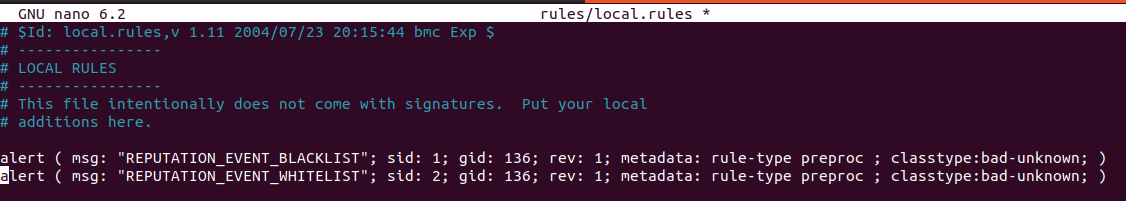




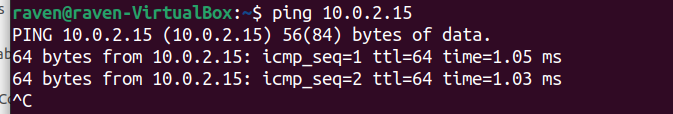
In black\_list.rules, we add the IP address of our known attacker.



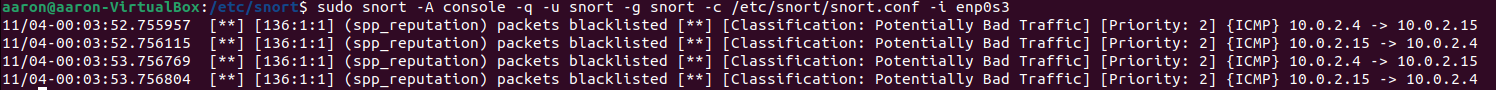
We add an additional line “scan\_local” to the reputation preprocessor in snorf.conf



And we also add some alerts to local.rules to trigger when whitelist and blacklist events occur.



After restarting snort with our newly implemented rules, we’ll test if it works. From our attacker, we send a ping to the victim.



We are notified of incoming traffic coming to and from blacklisted IP address.

References

<https://www.youtube.com/watch?v=iBsGSsbDMyw>

<https://www.researchgate.net/publication/338660054_DETECTING_DDoS_ATTACK_USING_Snort>

<https://resources.infosecinstitute.com/topic/icmp-attacks/>

<https://docs.snort.org/welcome>

<https://resources.infosecinstitute.com/topic/ethical-hacking-snmp-recon/>

<https://resources.infosecinstitute.com/topic/snmp-pentesting/>

<https://sublimerobots.com/2015/12/the-snort-reputation-preprocessor/>

<https://www.golinuxcloud.com/hping3-command-in-linux/>