Snort Lab

List of Contents

Installation & Configuration

Ubuntu Server 4GB

Bridged Network, Promiscuous mode "Allow All"

I SSH'ed into my Ubuntu VM with my Garuda distro

Configuration files:

la -al /etc/snort

sudo vim /etc/snort/snort.conf

Vim numbered:

`sudo vim /root/.vimrc`

```
set number
syntax on
```

We will remove all premade rules:

To comment multiple lines in vim:

`:597,717s/^/#`

Intrusion Detection with Snort

Configuring Rules:

`sudo vim /etc/snort/rules/local.rules`

Rules:

I want to generate an alert for any icmp traffic, that comes from any external address with any external port -> into our home network subnet ('sid' should have a unique value) and revisions for specific rules:

`alert icmp any any -> $HOME_NET any (msg: "ICMP Ping Detected!"; sid:100001; rev:1;)`

Running Snort:

man snort

-A - alert mode.

-l - log-dir, set the output logging directory to log-dir.

-i - interface

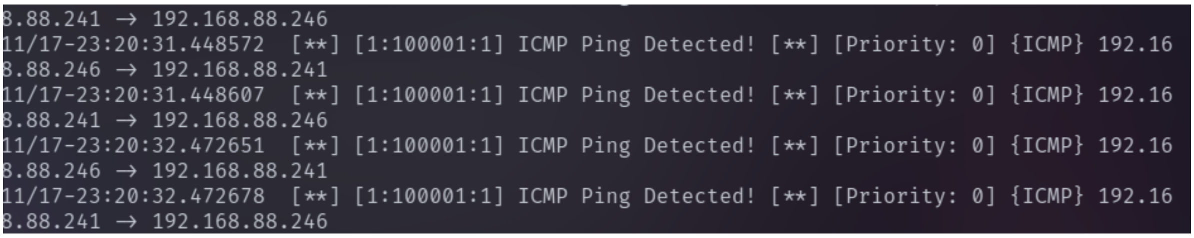
-q - running quietly, don't display banner and initialisation information.

Ping rule:

```
alert icmp any any -> $HOME_NET any (msg:"ICMP Ping Detected!"; sid:1000001; rev:1;)
```

Running Snort:

```
sudo snort -q -l /var/log/snort -i enp0s3 -A console -c /etc/snort/snort.conf
```



```
8.88.241 → 192.168.88.246
11/17-23:20:31.448572  [**] [1:100001:1] ICMP Ping Detected! [**] [Priority: 0] {ICMP} 192.16
8.88.246 → 192.168.88.241
11/17-23:20:31.448607  [**] [1:100001:1] ICMP Ping Detected! [**] [Priority: 0] {ICMP} 192.16
8.88.241 → 192.168.88.246
11/17-23:20:32.472651  [**] [1:100001:1] ICMP Ping Detected! [**] [Priority: 0] {ICMP} 192.16
8.88.246 → 192.168.88.241
11/17-23:20:32.472678  [**] [1:100001:1] ICMP Ping Detected! [**] [Priority: 0] {ICMP} 192.16
8.88.241 → 192.168.88.246
```

> Snort is giving alerts, when I ping the Ubuntu Desktop ^

SSH rule:

```
alert tcp any any -> $HOME_NET 22 (msg:"SSH Authentication Attemp!"; sid:1000002; rev:1;)
```

FTP Rule on a Specific IP/Device:

```
alert tcp any any -> 192.168.88.241 21 (msg:"FTP Authentication Attempt on Ubuntu_IDS"; sid:100003; rev:1; )
```

Using Inbuilt & Community Rules:

First, we have to uncomment the lines we commented earlier:

```
597,717s/^#//
```

Downloading the Snort Community 2.9 Rules from Snort's Website

I will copy the eternalblue exploit rule from the community rules:

```
alert tcp any any -> $HOME_NET 445 (msg:"OS-WINDOWS Microsoft Windows SMB remote code execution attempt"; flow:to_server,established; content:"|FF|SMB3|00 00 00 00|"; depth:9; offset:4; byte_extract:2,26,TotalDataCount,relative,little; byte_test:2,>,TotalDataCount,20,relative,little; metadata:policy balanced-ips drop, policy connectivity-ips drop, policy max-detect-ips drop, policy security-ips drop, ruleset community, service netbios-ssn; reference:cve,2017-0144; reference:cve,2017-0146; reference:url,blog.talosintelligence.com/2017/05/wannacry.html; reference:url,isc.sans.edu/forums/diary/ETERNALBLUE+Possible+Window+SMB+Buffer+Overflow+0Day/22304/; reference:url,technet.microsoft.com/en-us/security/bulletin/MS17-010; classtype:attempted-admin; sid:41978; rev:5;)
```

What we are telling Snort to do is alert us if there's any tcp connection coming from any network and port into our home network that connects to any device on our home network on port 445, whereby the flow is 'to server established' (the tcp state) and the content allows the user to set rules that search for specific content in the packet payload and trigger response based on that data.

Starting up the Windows 7 Machine...

We are using Metasploit to exploit the Windows 7 machine.

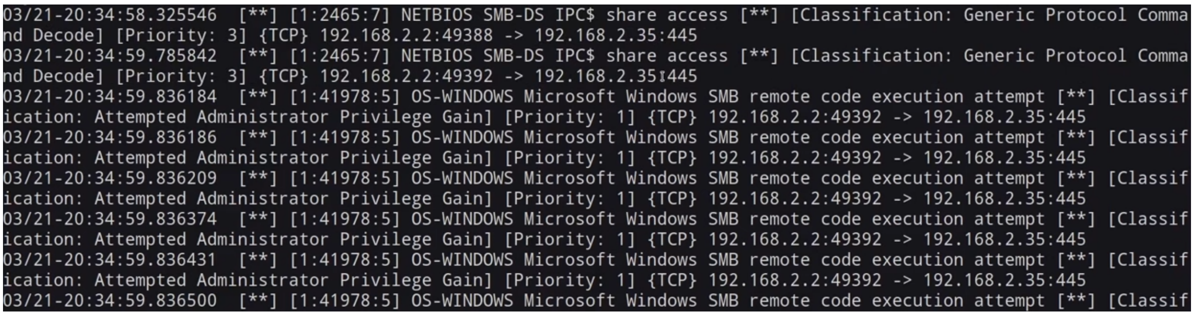
In Metasploit:

`use exploit /windows/smb/ms17_010_eternalblue`

`set RHOSTS 192.168.88.244` (Windows 7 VM IP)

`exploit`

Ubuntu IDS Machine:

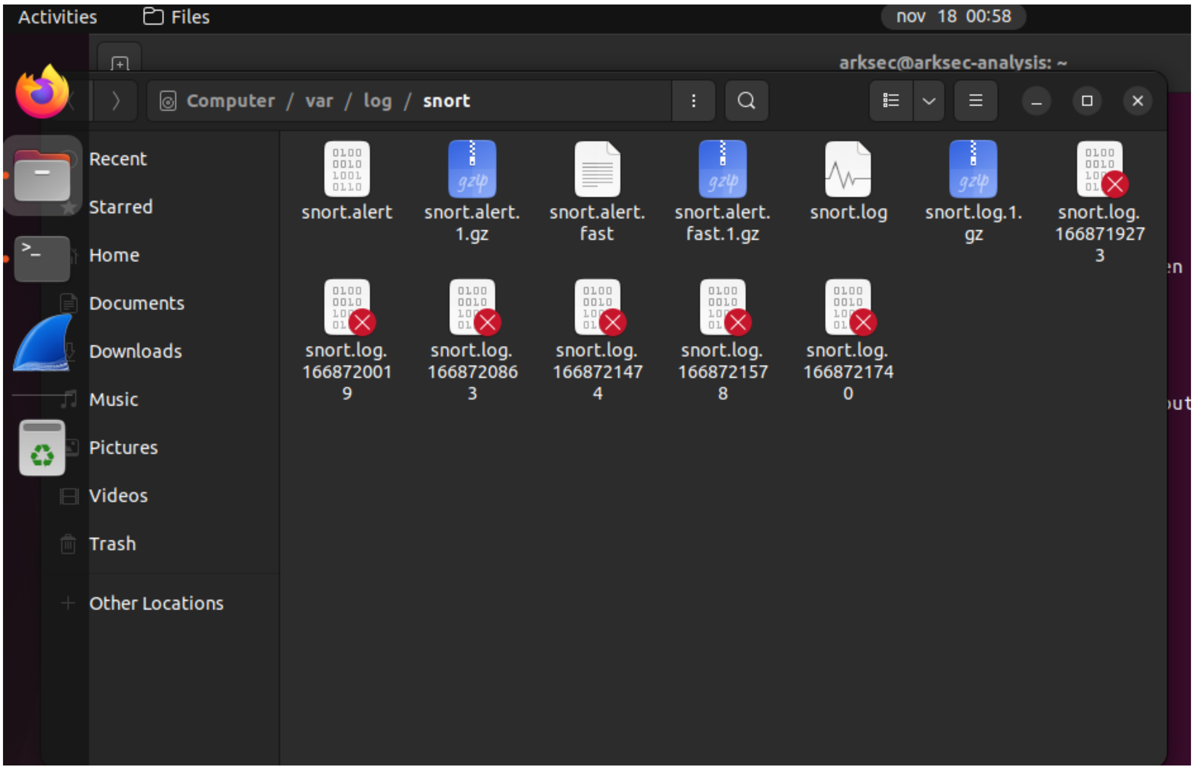


```
03/21-20:34:58.325546  [**] [1:2465:7] NETBIOS SMB-DS IPC$ share access [**] [Classification: Generic Protocol Comma
nd Decode] [Priority: 3] {TCP} 192.168.2.2:49388 -> 192.168.2.35:445
03/21-20:34:59.785842  [**] [1:2465:7] NETBIOS SMB-DS IPC$ share access [**] [Classification: Generic Protocol Comma
nd Decode] [Priority: 3] {TCP} 192.168.2.2:49392 -> 192.168.2.35:445
03/21-20:34:59.836184  [**] [1:41978:5] OS-WINDOWS Microsoft Windows SMB remote code execution attempt [**] [Classif
ication: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.2.2:49392 -> 192.168.2.35:445
03/21-20:34:59.836186  [**] [1:41978:5] OS-WINDOWS Microsoft Windows SMB remote code execution attempt [**] [Classif
ication: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.2.2:49392 -> 192.168.2.35:445
03/21-20:34:59.836209  [**] [1:41978:5] OS-WINDOWS Microsoft Windows SMB remote code execution attempt [**] [Classif
ication: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.2.2:49392 -> 192.168.2.35:445
03/21-20:34:59.836374  [**] [1:41978:5] OS-WINDOWS Microsoft Windows SMB remote code execution attempt [**] [Classif
ication: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.2.2:49392 -> 192.168.2.35:445
03/21-20:34:59.836431  [**] [1:41978:5] OS-WINDOWS Microsoft Windows SMB remote code execution attempt [**] [Classif
ication: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.2.2:49392 -> 192.168.2.35:445
03/21-20:34:59.836500  [**] [1:41978:5] OS-WINDOWS Microsoft Windows SMB remote code execution attempt [**] [Classif
```

Logging

If you take a look at the command that we used to run snort, then you can see that all logging was saved under /var/log/snort. However, the actual alerts were displayed on the console, so the only thing that was logged was the traffic, which is very important to understand.

`sudo snort -q -l /var/log/snort -i enp0s3 -A console -c /etc/snort/snort.conf`



These are all the log files that were created by Snort. This is essentially the raw packet capture on the network. This doesn't have any alerts attached to it and these can be accessed with Wireshark.

`sudo wireshark`

File > Open > under /var/log/snort, we open up the latest log.

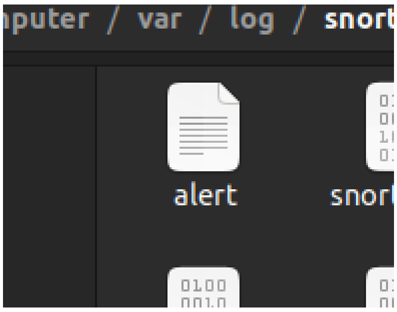
> Alerts are pretty much the most important piece of information, and if you are not logging them.. You should.

Creating a command, that logs everything

For the actual alert-mode, we are going to be using either fast, full or non. In the case of fast, this will write the alert to the default alert file in a single line syslog style. That is the preferred format if you are going to be using these logs to import them into tools like Splunk. In order to do that, instead of 'console', we are going to say 'fast' as an example:

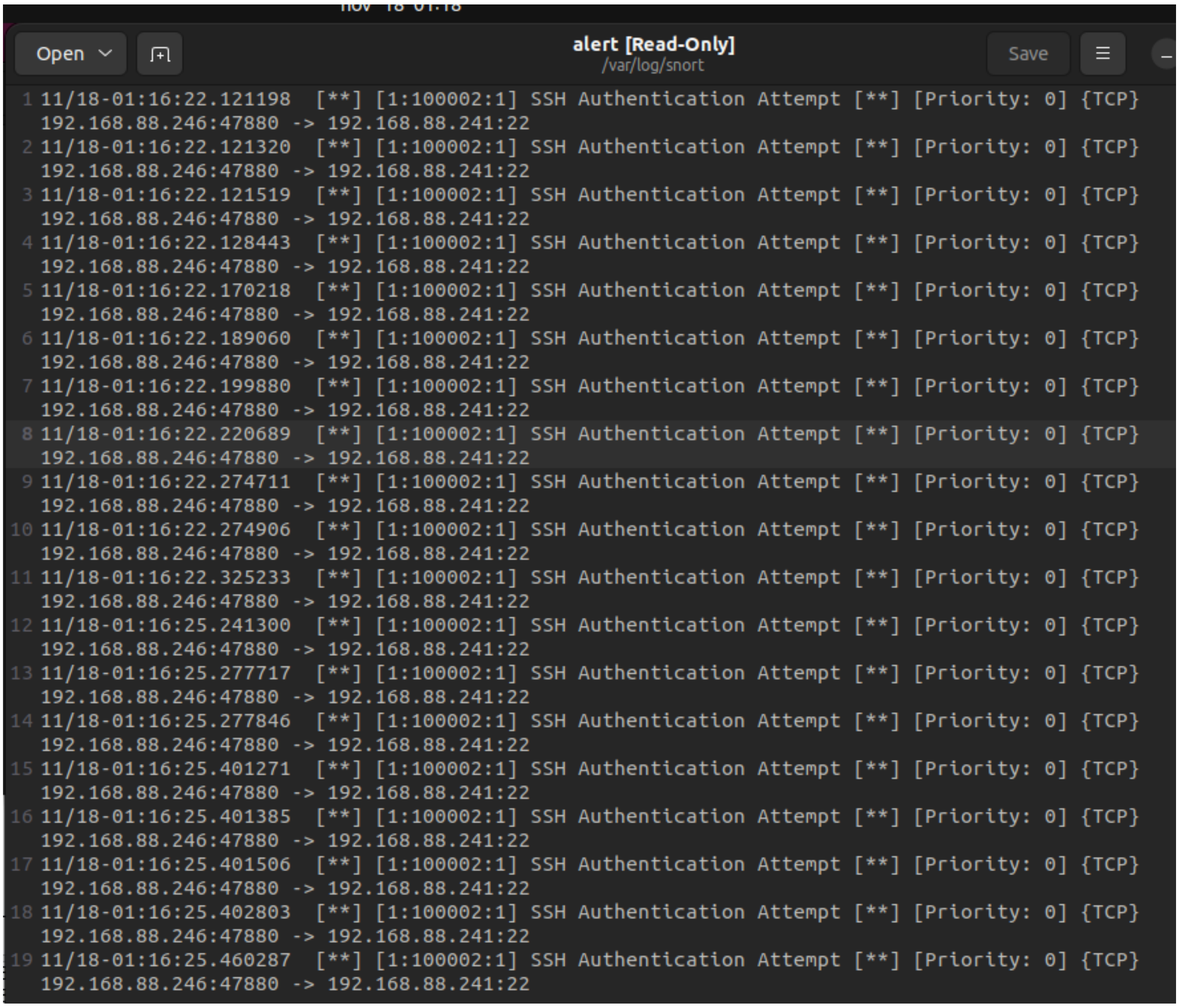
```
sudo snort -q -l /var/log/snort -i enp0s3 -A fast -c /etc/snort/snort.conf
```

> This will not display the alerts in the terminal.



The command generated the file 'alert' in the log folder.

The files content:



```
11/18-01:16:22.121198  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:22.121320  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:22.121519  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:22.128443  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:22.170218  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:22.189060  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:22.199880  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:22.220689  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:22.274711  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:22.274906  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:22.325233  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:25.241300  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:25.277717  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:25.277846  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:25.401271  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:25.401385  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:25.401506  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:25.402803  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
11/18-01:16:25.460287  [**] [1:100002:1] SSH Authentication Attempt [**] [Priority: 0] {TCP}
192.168.88.246:47880 -> 192.168.88.241:22
```

> We can import this format into tools like Splunk.

These logs can be analysed on this system or you can configure a log forwarder to essentially have the latest logs or alerts dynamically loaded onto a tool like Splunk for analysis, and at that point you'd be able to know what was happening on your network in regards to attacks and intrusions directly from Splunk.