

# INDEX

SR.NO	LISTS OF PRACTICAL	DATE	SIGN.
1	Encrypting and Decrypting Data Using OpenSSL	13/03/2024	
2	Demonstrate the use of Snort and Firewall Rules	16/03/2024	
3	Demonstrate Extract an Executable from a PCAP	20/03/2024	
4	Demonstrate Analysis of DNS Traffic	27/03/2024	
5	Create your own syslog Server	03/04/2024	
6	Configure your Linux system to send syslog messages to a syslog server and Read them	13/04/2024	
7	Install and Run Splunk on Linux	20/04/2024	
8	Install and Configure ELK on Linux	24/04/2024	
9	Install and Configure GrayLog on Linux	27/04/2024	
10	Demonstrate Conversion of Data into a Universal Format	04/05/2024	

## PRACTICAL 1

### Encrypting and Decrypting Data Using OpenSSL

#### Theory:

OpenSSL is an open source project that provides a robust, commercial-grade, and fullfeatured toolkit for the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols. It is also a general-purpose cryptography library. In this lab, you will use OpenSSL to encrypt and decrypt text messages.

**Note:** While OpenSSL is the de facto cryptography library today, the use presented in this lab is NOT recommended for robust protection. Below are two security problems with this lab:

1. The method described in this lab uses a weak key derivation function. The ONLY security is introduced by a very strong password.
2. The method described in this lab does not guarantee the integrity of the text file.

#### Required Resources

- CyberOps Workstation Virtual Machine
- Internet access

### Part 1: Encrypting Messages with OpenSSL

OpenSSL can be used as a standalone tool for encryption. While many encryption algorithms can be used, this lab focuses on AES. To use AES to encrypt a text file directly from the command line using OpenSSL, follow the steps below:

#### Step 1: Encrypting a Text File

- a. Log into CyberOPS Workstation VM.
- b. Open a terminal window.
- c. Because the text file to be encrypted is in the /home/analyst/lab.support.files/ directory, change to that directory:

```
[analyst@secOps ~]$ cd ./lab.support.files/  
[analyst@secOps lab.support.files]$
```

- d. Type the command below to list the contents of the encrypted **letter\_to\_grandma.txt** text file on the screen:

```
[analyst@secOps lab.support.files]$ cat letter_to_grandma.txt  
Hi Grandma,  
I am writing this letter to thank you for the chocolate chip cookies you sent
```

```
[analyst@secOps lab.support.files]$
```

- ```
[analyst@secOps lab.support.files]$
```

### Student choice of password

```
[analyst@secOps lab.support.files]$ cat message.enc
```

[illegible]

- g. To make the file readable, run the OpenSSL command again, but this time add the **a** option. The **-a** option tells OpenSSL to encode the encrypted message using a different encoding method of Base64 before storing the results in a file.

**Note:** Base64 is a group of similar binary-to-text encoding schemes used to represent binary data in an ASCII string format.

```
[analyst@secOps lab.support.files]$ openssl aes-256-cbc -a -in
letter_to_grandma.txt -out message.enc  enter aes-256-cbc
encryption password:
Verifying - enter aes-256-cbc encryption password:
```

- h. Once again, use the **cat** command to display the contents of the, now regenerated, **message.enc** file:

**Note:** The contents of **message.enc** will vary.

```
[analyst@secOps lab.support.files]$ cat message.enc
U2FsdGVkX19ApWyrn8RD5zNp0RPCuMGZ98wDc26u/vmj1zyDXobGQhm/dDRZasG7
rfnth5Q8NHValEw8vipKGM66dNFyyr9/hJUzCoqhFpRHgNn+Xs5+TOtz/QCPN1bi
08LGTSzOpfkg76XDck8uPy1hl/+Ng92sM5rgMzLXfEXtaYe5UgwOD42U/U6q73pj
a1ksQrTWsv5mtN7y6mh02Wobo3A1ooHrM7niOwK1a3YKrSp+ZhYzVTrtkswDI6Ci
XMufkv+FOGn+SoEEuh7l4fk0LIPEfGsExVFB4TGdTiZQApRw74rTAZaE/dopaJn0
sJmR3+3C+dmgzZIKEHWsJ2pgLvJ2Sme79J/XxwQVNpw=
[analyst@secOps lab.support.files]$
```

Is **message.enc** displayed correctly now? Explain.

Yes. While **message.enc** is encrypted, it is now correctly displayed because it has been converted from binary to text and encoded with Base64.

Can you think of a benefit of having **message.enc** Base64-encoded?

The encrypted message can now be copied and pasted in an email message, for example.

```
[analyst@secOps lab.support.files]$ openssl aes-256-cbc -a -in letter_to_grandma
.txt -out message.enc
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
[analyst@secOps lab.support.files]$ cat message.enc
U2FsdGVkX1+ooB/vrqH1B2lcAGSEzSou3GK/uQWBiljinzog4jR5HDborbrSgTFi
SF0Us4jflUfDnZgXGQNE/uk1hGna70VRo51YkcaFFCGAHLs5SpWeh50ML80JYy90
99IbPvC9iZmf+q6QLvUHOFFIyYgiNku/Ldt3buyH60XPeVC6D6AExTWzVvHnsn2s
ev5eixsKTU7UrVEXwD1kF1b1KwZakEzTGz6xV88E3cqBQbJRtbp700/hNhYQR00E
iV1VcpX3Xyh1FCUSC3m2LuZcAPtrKuS3aIcbIVJC1ZuX/iHsZD5eveDML+/C+B7w
Qy0KGjXWaNj3zQj4HYJZLSdD47DPXDblwhFYmP1mdp8=
[analyst@secOps lab.support.files]$
```

## Part 2: Decrypting Messages with OpenSSL

With a similar OpenSSL command, it is possible to decrypt **message.enc**.

- a. Use the command below to decrypt **message.enc**:

```
[analyst@secOps lab.support.files]$ openssl aes-256-cbc -a -d -in message.enc -out decrypted_letter.txt
```

- b. OpenSSL will ask for the password used to encrypt the file. Enter the same password again.
- c. When OpenSSL finishes decrypting the **message.enc** file, it saves the decrypted message in a text file called **decrypted\_letter.txt**. Use the **cat** display the contents of **decrypted\_letter.txt**:

```
[analyst@secOps lab.support.files]$ cat decrypted_letter.txt
```

Was the letter decrypted correctly?

Yes, the letter was decrypted correctly. The command used to decrypt also contains -a option.

Can you explain? Because message.enc was Base64 encoded after the encryption process took place, message.enc must be Base64 decoded before OpenSSL can decrypt it

```
[analyst@secOps lab.support.files]$ openssl aes-256-cbc -a -d -in message.enc -out decrypted_letter.txt
enter aes-256-cbc decryption password:
[analyst@secOps lab.support.files]$ cat decrypted_letter.txt
Hi Grandma,
I am writing this letter to thank you for the chocolate chip cookies you sent me. I
got them this morning and I have already eaten half of the box! They are absolutely
delicious!

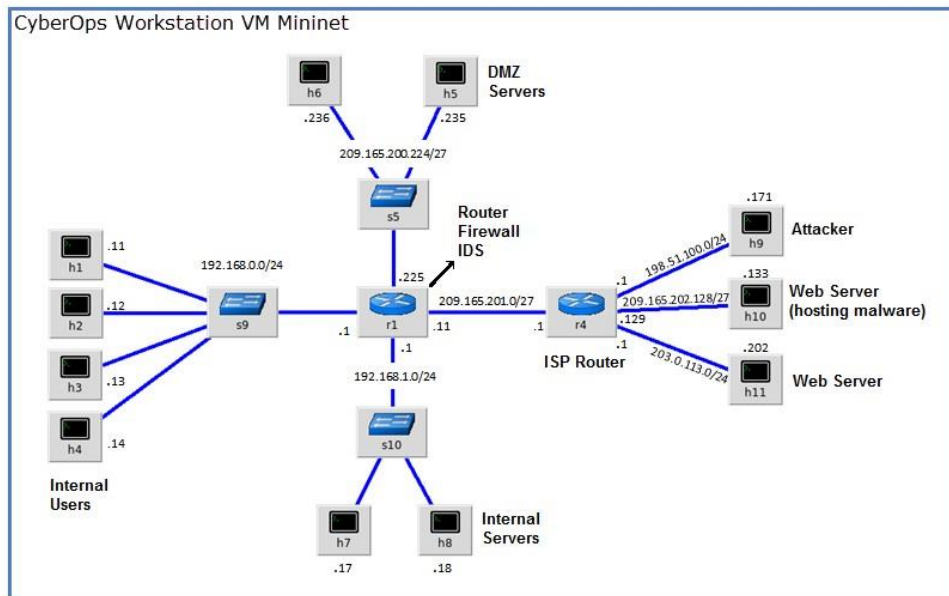
I wish you all the best. Love,
Your cookie-eater grandchild.

[analyst@secOps lab.support.files]$
```

## PRACTICAL 2

### Demonstrate the use of Snort and Firewall Rules

Topology:



Objectives:

- Part 1: Preparing the Virtual Environment
- Part 2: Firewall and IDS Logs
- Part 3: Terminate and Clear Mininet Process

Theory:

In a secure production network, network alerts are generated by various types of devices such as security appliances, firewalls, IPS devices, routers, switches, servers, and more. The problem is that not all alerts are created equally. For example, alerts generated by a server and alerts generated by a firewall will be different and vary in content and format.

### Steps:

#### Part 1: Preparing the Virtual Environment

a. Launch **Oracle VirtualBox** and change the **CyberOps Workstation** for Bridged mode, if necessary. Select **Machine > Settings > Network**. Under **Attached To**, select **Bridged Adapter** (or if you are using WiFi with a proxy, you may need **NAT adapter**) and click **OK**

b. Launch the **CyberOps Workstation VM**, open a terminal and configure its network by executing the **sh** script.

Because the script requires super-user privileges, provide the password for the user **analyst**.

```
[analyst@secOps ~]$ sudo ./lab.support.files/scripts/configure_as_dhcp.sh [sudo]
password for analyst:
[analyst@secOps ~]$
```

c. Use the **ifconfig** command to verify **CyberOps Workstation VM** now has an IP address on your local network. You can also test connectivity to a public webserver by pinging **www.cisco.com**. Use **Ctrl+C** to stop the pings.

```
[analyst@secOps ~]$ ping www.cisco.com
PING e2867.dsca.akamaiedge.net (23.204.15.199) 56(84) bytes of data.

64 bytes from a23-204-15-199.deploy.static.akamaitechnologies.com (23.204.15.199): icmp_seq=1 ttl=54 time=28.4 ms

64 bytes from a23-204-15-199.deploy.static.akamaitechnologies.com (23.204.15.199): icmp_seq=2 ttl=54 time=35.5 ms

^C

--- e2867.dsca.akamaiedge.net ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms rtt
min/avg/max/mdev = 28.446/32.020/35.595/3.578 ms
```

```
icmp_seq=12 ttl=57 time=4.76 ms
64 bytes from a23-53-69-59.deploy.static.akamaitechnologies.com (23.53.69.59): icmp_seq=13 ttl=57 time=11.6 ms
64 bytes from a23-53-69-59.deploy.static.akamaitechnologies.com (23.53.69.59): icmp_seq=14 ttl=57 time=4.91 ms
^C
--- e2867.dsca.akamaiedge.net ping statistics ---
14 packets transmitted, 14 received, 0% packet loss, time 13023ms
rtt min/avg/max/mdev = 4.327/6.435/14.686/3.033 ms
[analyst@secOps ~]$
```

## Part 2: Firewall and IDS Logs

Firewalls and Intrusion Detection Systems (IDS) are often deployed to partially automate the traffic monitoring task. Both firewalls and IDSs match incoming traffic against administrative rules. Firewalls usually compare the packet header against a rule set while IDSs often use the packet payload for rule set comparison. Because firewalls and IDSs apply the pre-defined rules to different portions of the IP packet, IDS and firewall rules have different structures.

While there is a difference in rule structure, some similarities between the components of the rules remain. For example, both firewall and IDS rules contain matching components and action components. Actions are taken after a match is found.

- **Matching component** – specifies the packet elements of interest, such as: packet source; the packet destination; transport layer protocols and ports; and data included in the packet payload.
- **Action component** – specifies what should be done with that packet that matches a component, such as: accept and forward the packet; drop the packet; or send the packet to a secondary rule set for further inspection.

A common firewall design is to drop packets by default while manually specifying what traffic should be allowed. Known as dropping-by-default, this design has the advantage protecting the network from unknown protocols and attacks. As part of this design, it is common to log the events of dropped packets since these are packets that were not explicitly allowed and therefore, infringe on the organization's policies. Such events should be recorded for future analysis.

### Step 1: Real-Time IDS Log Monitoring

- a.** From the **CyberOps Workstation VM**, run the script to start **mininet**.(pass: cyberops)

```
[analyst@secOps ~]$ sudo ./lab.support.files/scripts/cyberops_extended_topo_no_fw.py [sudo]
password for analyst:
*** Adding controller
*** Add switches
*** Add hosts
*** Add links
*** Starting network
*** Configuring hosts
R1 R4 H1 H2 H3 H4 H5 H6 H7 H8 H9 H10 H11
*** Starting controllers
*** Starting switches
*** Add routes
*** Post configure switches and hosts ***
Starting CLI:
mininet>
```

The **mininet** prompt should be displayed, indicating **mininet** is ready for commands.

- b.** From the **mininet** prompt, open a shell on **R1** using the command below:

```
mininet> xterm R1 mininet>
```

The **R1** shell opens in a terminal window with black text and white background.

What user is logged into that shell? What is the indicator of this?

The root user. This is indicated by the # sign after the prompt. **c.**

From **R1's** shell, start the Linux-based IDS, Snort.



```
[root@secOps analyst]# ./lab.support.files/scripts/start_snort.sh
```

```
Running in IDS mode --==
```

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

```
Initializing Output Plugins!
```

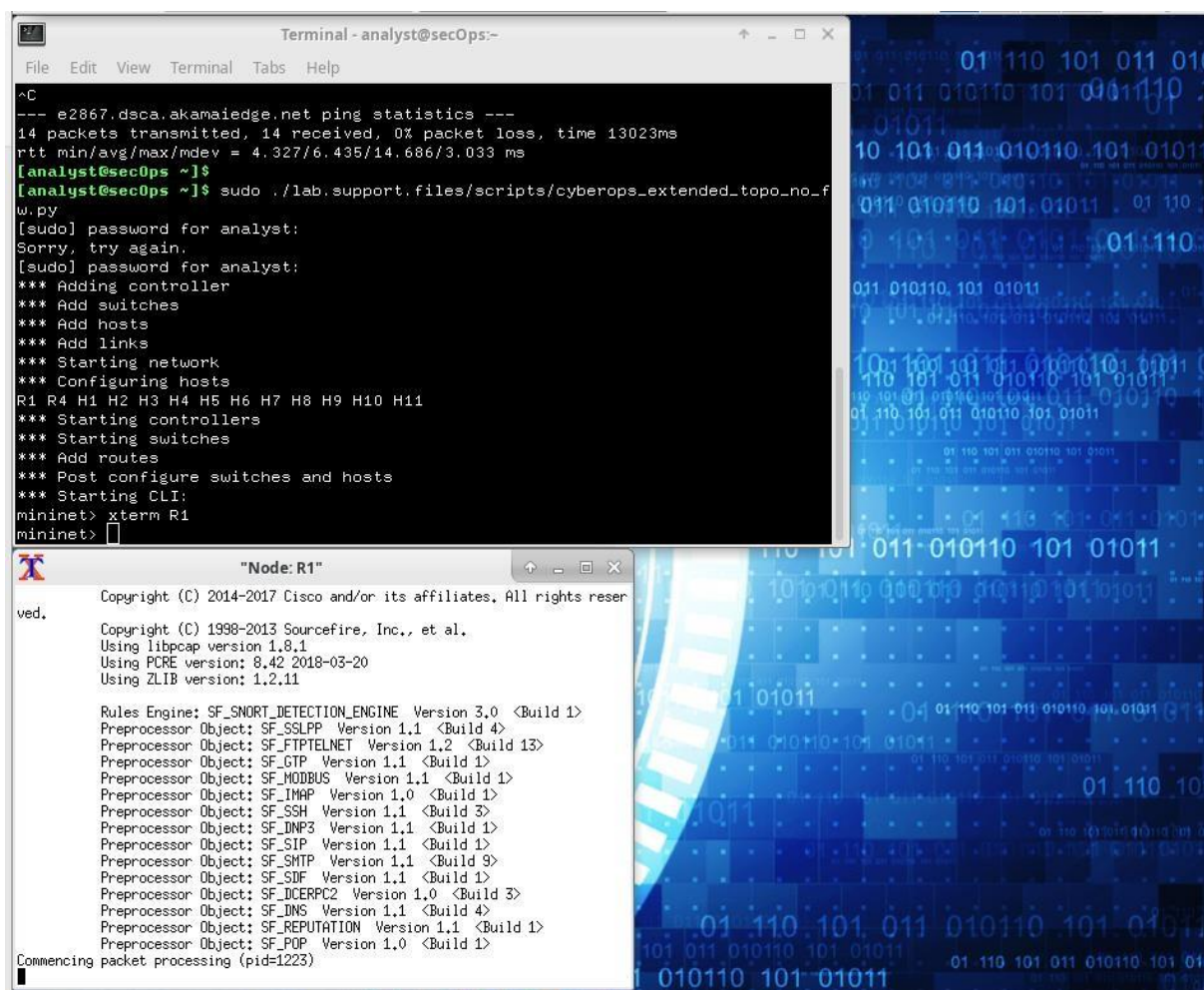
```
Initializing Preprocessors!
```

```
Initializing Plug-ins!
```

```
Parsing Rules file "/etc/snort/snort.conf"
```

```
<output omitted>
```

**Note:** You will not see a prompt as Snort is now running in this window. If for any reason, Snort stops running and the `[root@secOps analysts]#` prompt is displayed, rerun the script to launch Snort. Snort must be running to capture alerts later in the lab.



```
Terminal - analyst@secOps:-
File Edit View Terminal Tabs Help

^C
--- e2867.dsca.akamaiedge.net ping statistics ---
14 packets transmitted, 14 received, 0% packet loss, time 13023ms
rtt min/avg/max/mdev = 4.327/6.435/14.686/3.033 ms
[analyst@secOps ~]$
[analyst@secOps ~]$ sudo ./lab.support.files/scripts/cyberops_extended_topo_no_fw.py
[sudo] password for analyst:
Sorry, try again.
[sudo] password for analyst:
*** Adding controller
*** Add switches
*** Add hosts
*** Add links
*** Starting network
*** Configuring hosts
R1 R4 H1 H2 H3 H4 H5 H6 H7 H8 H9 H10 H11
*** Starting controllers
*** Starting switches
*** Add routes
*** Post configure switches and hosts
*** Starting CLI:
mininet> xterm R1
mininet>
```

```
"Node: R1"
Copyright (C) 2014-2017 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.8.1
Using PCRE version: 8.42 2018-03-20
Using ZLIB version: 1.2.11

Rules Engine: SF_SNORT_DETECTION_ENGINE Version 3.0 <Build 1>
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
Preprocessor Object: SF_INMAP Version 1.0 <Build 1>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_DICERPC2 Version 1.0 <Build 3>
Preprocessor Object: SF_DNS Version 1.1 <Build 4>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
Preprocessor Object: SF_POP Version 1.0 <Build 1>

Commencing packet processing (pid=1223)
```

d. From the **CyberOps Workstation VM mininet** prompt, open shells for hosts **H5** and **H10**.

```
mininet> xterm H5 mininet>
xterm H10 mininet>
```

e. **H10** will simulate a server on the Internet that is hosting malware. On **H10**, run the **mal\_server\_start.sh** script to start the server.

```
[root@secOps analyst]# ./lab.support.files/scripts/mal_server_start.sh
[root@secOps analyst]#
```

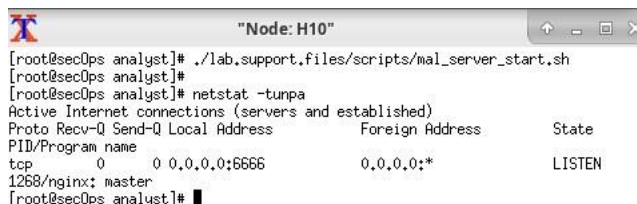
f. On **H10**, use **netstat** with the **-tunpa** options to verify that the web server is running. When used as shown below, **netstat** lists all ports currently assigned to services:

```
[root@secOps analyst]# netstat -tunpa

Active Internet connections (servers and established)

Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name    tcp
0      0 0.0.0.0:6666             0.0.0.0:*               LISTEN      1839/nginx: master  [root@secOps analyst]#
```

As seen by the output above, the lightweight webserver **nginx** is running and listening to connections on port TCP 6666.



```
"Node: H10"
[root@secOps analyst]# ./lab.support.files/scripts/mal_server_start.sh
[root@secOps analyst]#
[root@secOps analyst]# netstat -tunpa
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp      0      0 0.0.0.0:6666             0.0.0.0:*               LISTEN
1268/nginx: master
[root@secOps analyst]#
```

g. In the **R1** terminal window, an instance of Snort is running. To enter more commands on **R1**, open another **R1** terminal by entering the **xterm R1** again in the **CyberOps Workstation VM** terminal window. You may also want to arrange the terminal windows so that you can see and interact with each device.

h. In the new **R1** terminal tab, run the **tail** command with the **-f** option to monitor the **/var/log/snort/alert** file in real-time. This file is where snort is configured to record alerts.

```
[root@secOps analyst]# tail -f /var/log/snort/alert
```

Because no alerts were yet recorded, the log should be empty. However, if you have run this lab before, old alert entries may be shown. In either case, you will not receive a prompt after typing this command. This window will display alerts as they happen.

i. From **H5**, use the **wget** command to download a file named **Nimda.Amm.exe**. Designed to download content via HTTP, **wget** is a great tool for downloading files from web servers directly from the command line.

```
[root@secOps analyst]# wget 209.165.202.133:6666/W32.Nimda.Amm.exe -
-2017-04-28 17:00:04-- http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... connected.
HTTP request sent, awaiting response... 200 OK
Length: 345088 (337K) [application/octet-stream]
Saving to: 'W32.Nimda.Amm.exe'

W32.Nimda.Amm.exe  100%[=====>] 337.00K  --.
-KB/s  in 0.02s

2017-04-28 17:00:04 (16.4 MB/s) - 'W32.Nimda.Amm.exe' saved [345088/345088]

[root@secOps analyst]#
```

What port is used when communicating with the malware web server? What is the indicator?

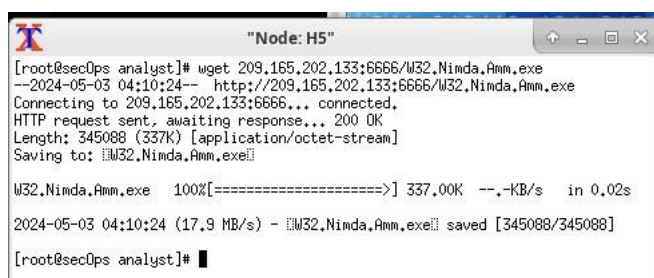
Port 6666. The port was specified in the URL, after the : separator.

Was the file completely downloaded?

Yes

Did the IDS generate any alerts related to the file download?

Yes



```
"Node: H5"
[root@secOps analyst]# wget 209.165.202.133:6666/W32.Nimda.Amm.exe
--2024-05-03 04:10:24-- http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... connected.
HTTP request sent, awaiting response... 200 OK
Length: 345088 (337K) [application/octet-stream]
Saving to: 'W32.Nimda.Amm.exe'

W32.Nimda.Amm.exe  100%[=====>] 337.00K  --.-KB/s  in 0.02s

2024-05-03 04:10:24 (17.9 MB/s) - 'W32.Nimda.Amm.exe' saved [345088/345088]

[root@secOps analyst]#
```

j. As the malicious file was transiting **R1**, the IDS, Snort, was able to inspect its payload. The payload matched at least one of the signatures configured in Snort and triggered an alert on the second **R1** terminal window (the tab where **tail -f** is running). The alert entry is shown below. Your timestamp will be different:

```
04/28-17:00:04.092153 [*] [1:1000003:0] Malicious Server Hit! [*] [Priority: 0] {TCP} 209.165.200.235:34484 -> 209.165.202.133:6666
```

Based on the alert shown above, what were the source and destination IPv4 addresses used in the transaction?

Source IP: 209.165.200.235; Destination IP: 209.165.202.133.

Based on the alert shown above, what was the source and destination ports used in the transaction?

Source port: 34484; Destination port: 6666. (Note: the source port will vary).

Based on the alert shown above, when did the download take place?

April 28th around 5pm for the example, but the student's answer will be different.

Based on the alert shown above, what was the message recorded by the IDS signature?

“Malicious Server Hit!”

On **H5**, use the `tcpdump` command to capture the event and download the malware file again so you can capture the transaction. Issue the following command below start the packet capture:

```
[root@secOps analyst]# tcpdump -i H5-eth0 -w nimda.download.pcap &
[1] 5633

[root@secOps analyst]# tcpdump: listening on H5-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

The command above instructs `tcpdump` to capture packets on interface **H5-eth0** and save the capture to a file named **nimda.download.pcap**.

The **&** symbol at the end tells the shell to execute **tcpdump** in the background. Without this symbol, **tcpdump** would make the terminal unusable while it was running. Notice the **[1] 5633**; it indicates one process was sent to background and its process ID (PID) is 5366. Your PID will most likely be different.

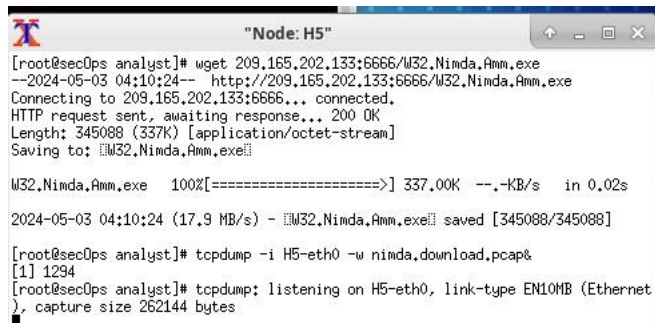
**k.** Press **ENTER** a few times to regain control of the shell while **tcpdump** runs in background.

**l.** Now that **tcpdump** is capturing packets, download the malware again. On **H5**, rerun the command or use the up arrow to recall it from the command history facility.

```
[root@secOps analyst]# wget 209.165.202.133:6666/W32.Nimda.Amm.exe -
-2017-05-02 10:26:50-- http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... connected.
HTTP request sent, awaiting response... 200 OK
Length: 345088 (337K) [application/octet-stream]
Saving to: 'W32.Nimda.Amm.exe'

W32.Nimda.Amm.exe 100%[=====>] 337.00K --.-KB/s in 0.003s

2017-05-02 10:26:50 (105 MB/s) - 'W32.Nimda.Amm.exe' saved [345088/345088]
```



```

[Node: H5]
[root@secOps analyst]# wget 209.165.202.133:6666/W32.Nimda.Amm.exe
--2024-05-03 04:10:24-- http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... connected.
HTTP request sent, awaiting response... 200 OK
Length: 345088 (337K) [application/octet-stream]
Saving to: 'W32.Nimda.Amm.exe'

W32.Nimda.Amm.exe 100%[=====] 337.00K --.-KB/s in 0.02s

2024-05-03 04:10:24 (17.9 MB/s) - 'W32.Nimda.Amm.exe' saved [345088/345088]

[root@secOps analyst]# tcpdump -i H5-eth0 -w nimda.download.pcap&
[1] 1294
[root@secOps analyst]# tcpdump: listening on H5-eth0, link-type EN10MB (Ethernet
), capture size 262144 bytes

```

m. Stop the capture by bringing **tcpdump** to foreground with the **fg** Because **tcpdump** was the only process sent to background, there is no need to specify the PID. Stop the **tcpdump** process with **Ctrl+C**. The **tcpdump** process stops and displays a summary of the capture. The number of packets may be different for your capture.

```

[root@secOps analyst]# fg
tcpdump -i h5-eth0 -w nimda.download.pcap
^C316 packets captured
316 packets received by filter
0 packets dropped by kernel
[root@secOps analyst]#

```

```

└─┬
[root@secOps analyst]# fg
tcpdump -i H5-eth0 -w nimda.download.pcap
^C0 packets captured
0 packets received by filter
0 packets dropped by kernel
[root@secOps analyst]#

```

n. On **H5**, Use the **ls -l** command to verify the pcap file was in fact saved to disk and has size greater than zero:

```

[root@secOps analyst]# ls -l
total 1400
drwxr-xr-x 2 analyst analyst 4096 Sep 26 2014 Desktop drwx--
---- 3 analyst analyst 4096 Jul 14 11:28 Downloads drwxr-xr-x
8 analyst analyst 4096 Jul 25 16:27 lab.support.files -rw-r--r-- 1
root root 371784 Aug 17 14:48 nimda.download.pcap drwxr-
xr-x 2 analyst analyst 4096 Mar 3 15:56 second_drive
-rw-r--r-- 1 root root 345088 Apr 14 15:17
W32.Nimda.Amm.exe
-rw-r--r-- 1 root root 345088 Apr 14 15:17
W32.Nimda.Amm.exe.l
[root@secOps analyst]#

```

**Note:** Your directory list may have a different mix of files, but you should still see the **nimda.download.pcap** file.

How can be this PCAP file be useful to the security analyst?

PCAP files contain the packets related to the traffic seen by the capturing NIC. In that way, the PCAP is very useful to re-retrace network events such as communication to malicious end points. Tools such as Wireshark can be used to facilitate PCAP analysis.

**Note:** The analysis of the PCAP file will be performed in another lab.

```
[root@secOps analyst]# ls -l
total 360
drwxr-xr-x 2 analyst analyst 4096 Mar 22 2018 Desktop
drwxr-xr-x 3 analyst analyst 4096 Mar 22 2018 Downloads
drwxr-xr-x 9 analyst analyst 4096 May 3 03:53 lab.support.files
-rw-r--r-- 1 root root 24 May 3 04:15 ninda.download.pcap
drwxr-xr-x 2 analyst analyst 4096 Mar 21 2018 second_drive
-rw-r--r-- 1 root root 345088 Mar 23 2018 W32.Ninda.Amm.exe
[root@secOps analyst]#
```

## Step 2: Tuning Firewall Rules Based on IDS Alerts

In Step 1, you started an internet-based malicious server. To keep other users from reaching that server, it is recommended to block it in the edge firewall.

In this lab's topology, **R1** is not only running an IDS but also a very popular Linuxbased firewall called **iptables**. In this step, you will block traffic to the malicious server identified in Step 1 by editing the firewall rules currently present in **R1**.

**Note:** While a comprehensive study of **iptables** is beyond the scope of this course, **iptables** basic logic and rule structure is fairly straight-forward.

The firewall **iptables** uses the concepts of *chains* and *rules* to filter traffic.

Traffic entering the firewall and destined to the firewall device itself is handled by the **INPUT** chain. Examples of this traffic are ping packets coming from any other device on any networks and sent to any one of the firewall's interfaces.

Traffic originated in the firewall device itself and destined to somewhere else, is handled by the **OUTPUT** chain. Examples of this traffic are ping responses generated by the firewall device itself.

Traffic originated somewhere else and passing through the firewall device is handled by the **FORWARD** chain. Examples of this traffic are packets being routed by the firewall.

Each chain can have its own set of independent rules specifying how traffic is to be filtered for that chain. A chain can have practically any number of rules, including no rule at all.

Rules are created to check specific characteristics of packets, allowing administrators to create very comprehensive filters. If a packet doesn't match a rule, the firewall moves on to the next rule and checks again. If a match is found, the firewall takes the action defined in the matching rule. If all rules in a chain have been checked and yet no match was found, the firewall takes the action specified in the chain's policy, usually allow the packet to flow through or deny it.

**a.** In the **CyberOps Workstation VM**, start a third R1 terminal window.

```
mininet > xterm R1
```

b. In the new **R1** terminal window, use the `iptables` command to list the chains and their rules currently in use:

```
[root@secOps ~]# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination

Chain FORWARD (policy ACCEPT 6 packets, 504 bytes)
 pkts bytes target    prot opt in     out     source                   destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination

[root@secOps ~]#
```

What chains are currently in use by **R1**?  
**INPUT, OUTPUT and FORWARD**

```
[root@secOps analyst]# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination

[root@secOps analyst]#
```

c. Connections to the malicious server generate packets that must transverse the **iptables** firewall on **R1**. Packets traversing the firewall are handled by the **FORWARD** rule and therefore, that is the chain that will receive the blocking rule. To keep user computers from connecting to the malicious server identified in Step 1, add the following rule to the **FORWARD** chain on **R1**:

```
[root@secOps ~]# iptables -I FORWARD -p tcp -d 209.165.202.133 --dport 6666 -j DROP
[root@secOps ~]#
```

Where:

- **-I FORWARD**: inserts a new rule in the **FORWARD** chain.
- **-p tcp**: specifies the TCP protocol.
- **-d 209.165.202.133**: specifies the packet's destination • **--dport 6666**: specifies the destination port • **-j DROP**: set the action to drop.

d. Use the `iptables` command again to ensure the rule was added to the **FORWARD** chain. The CyberOps Workstation VM may take a few seconds to generate the output:



```
[root@secOps analyst]# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
0 0 DROP tcp -- any any anywhere 209.165.202.133 tcp dpt:6666

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination

[root@secOps analyst]# iptables -I FORWARD -p tcp -d 209.165.202.133 --dport 66
66 -j DROP
[root@secOps analyst]# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
0 0 DROP tcp -- any any anywhere 209.165.202.
133 tcp dpt:6666

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination

[root@secOps analyst]#
```

e. On H5, try to download the file again:

```
[root@secOps analyst]# wget 209.165.202.133:6666/W32.Nimda.Amm.exe
--2017-05-01 14:42:37-- http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... failed: Connection timed out.
Retrying.

--2017-05-01 14:44:47-- (try: 2) http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... failed: Connection timed out.
Retrying.
```

Enter **Ctrl+C** to cancel the download, if necessary.

Was the download successful this time? Explain.

No. The firewall is blocking connections to the malware hosting server.

What would be a more aggressive but also valid approach when blocking the offending server?

Instead of specifying IP, protocol and port, a rule could simply block the server's IP address. This would completely cut access to that server from the internal network.



### Part 3: Terminate and Clear Mininet Process

- a. Navigate to the terminal used to start Mininet. Terminate the Mininet by entering **quit** in the main CyberOps VM terminal window.
- b. After quitting Mininet, clean up the processes started by Mininet. Enter the password **cyberops** when prompted.

```
[analyst@secOps scripts]$ sudo mn -c  
[sudo] password for analyst:
```

```
mininet> sudo mn -c  
*** Unknown command: sudo mn -c  
mininet> exit  
*** Stopping 0 controllers  
  
*** Stopping 4 terms  
*** Stopping 15 links  
.....  
*** Stopping 3 switches  
S5 S9 S10  
*** Stopping 13 hosts  
R1 R4 H1 H2 H3 H4 H5 H6 H7 H8 H9 H10 H11  
*** Done  
[analyst@secOps ~]$ sudo mn -c  
[sudo] password for analyst:  
*** Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes  
killall controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openflowd  
ovs-controller udpbwtest mnexec ixs 2> /dev/null  
killall -9 controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openfl
```

## PRACTICAL 3

### Demonstrate Extract an Executable from a PCAP

Objectives:

Part 1: Analyze Pre-Captured Logs and Traffic Captures

Part 2: Extract Downloaded Files from PCAP

Theory:

Looking at logs is very important, but it is also important to understand how network transactions happen at the packet level.

In this lab, you will analyze the traffic in a previously captured pcap file and extract an executable from the file. **Steps:**

#### Part 1: Analyze Pre-Captured Logs and Traffic Captures

In Part 2, you will work with the nimda.download.pcap file. Captured in a previous lab, nimda.download.pcap contains the packets related to the download of the Nimda malware. Your version of the file, if you created it in the previous lab and did not reimport your CyberOps Workstation VM, is stored in the /home/analyst directory. However, a copy of that file is also stored in the CyberOps Workstation VM, under the /home/analyst/lab.support.files/pcaps directory so that you can complete this lab. For consistency of output, the lab will use the stored version in the pcaps directory.

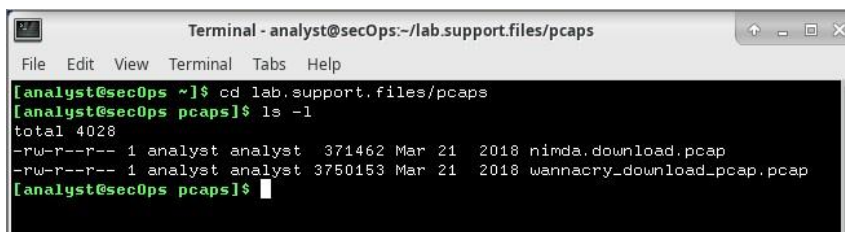
While tcpdump can be used to analyze captured files, Wireshark's graphical interface makes the task much easier. It is also important to note that tcpdump and Wireshark share the same file format for packet captures; therefore, PCAP files created by one tool can be opened by the other.

1. Change directory to the lab.support.files/pcaps folder, and get a listing of files using the ls -l command.

```
[analyst@secOps ~]$ cd lab.support.files/pcaps
```

```
[analyst@secOps pcaps]$ ls -l total
```

```
7460
```

A terminal window titled "Terminal - analyst@secOps:~/lab.support.files/pcaps" showing the execution of 'cd lab.support.files/pcaps' and 'ls -l'. The output of 'ls -l' shows three files: 'nimda.download.pcap' (371462 bytes, Mar 21, 2018), 'wannacry\_download.pcap.pcap' (3750153 bytes, Mar 21, 2018), and 'lab\_prep.pcap' (3510551 bytes, Aug 7, 2018). The prompt is currently at the end of the third line.

```
Terminal - analyst@secOps:~/lab.support.files/pcaps
File Edit View Terminal Tabs Help
[analyst@secOps ~]$ cd lab.support.files/pcaps
[analyst@secOps pcaps]$ ls -l
total 4028
-rw-r--r-- 1 analyst analyst 371462 Mar 21 2018 nimda.download.pcap
-rw-r--r-- 1 analyst analyst 3750153 Mar 21 2018 wannacry_download.pcap.pcap
[analyst@secOps pcaps]$
```

```
-rw-r--r-- 1 analyst analyst 3510551 Aug 7 15:25 lab_prep.pcap
```

```
-rw-r--r-- 1 analyst analyst 371462 Jun 22 10:47
```

nimda.download.pcap

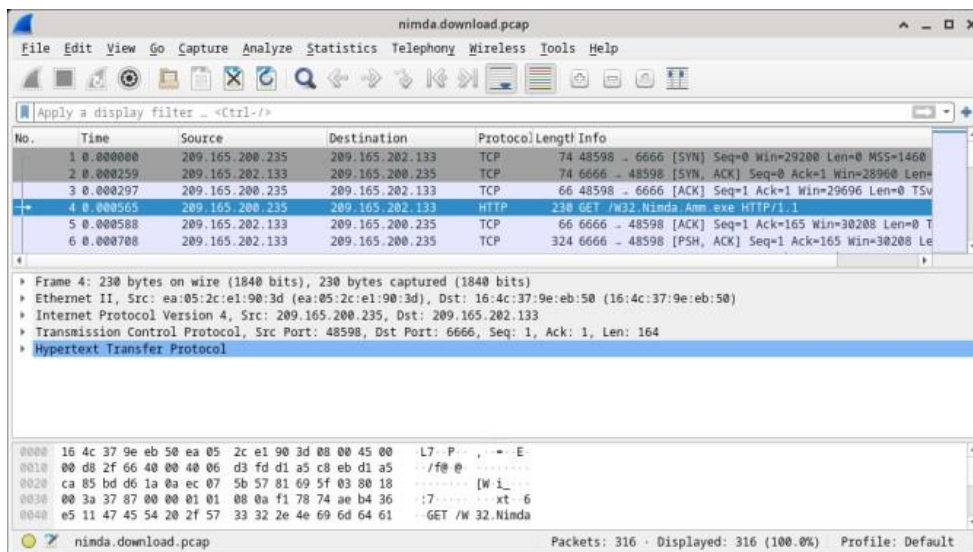
```
-rw-r--r-- 1 analyst analyst 3750153 May 25 11:10 wannacry_download_pcap.pcap
```

```
[analyst@secOps pcaps]$
```

2. Issue the command below to open the nimda.download.pcap file in Wireshark.

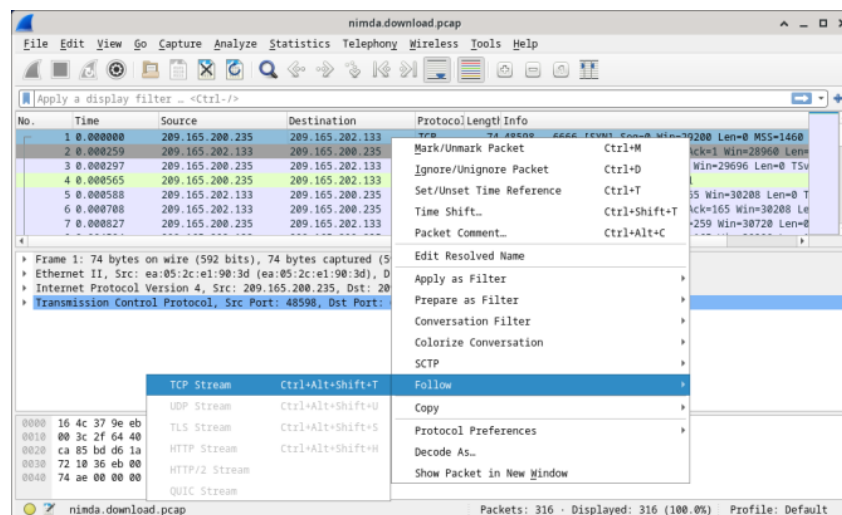
```
[analyst@secOps pcaps]$ wireshark nimda.download.pcap &
```

3. The nimda.download.pcap file contains the packet capture related to the malware download performed in a previous lab. The pcap contains all the packets sent and received while tcpdump was running. Select the fourth packet in the capture and expand the Hypertext Transfer Protocol to display as shown below.

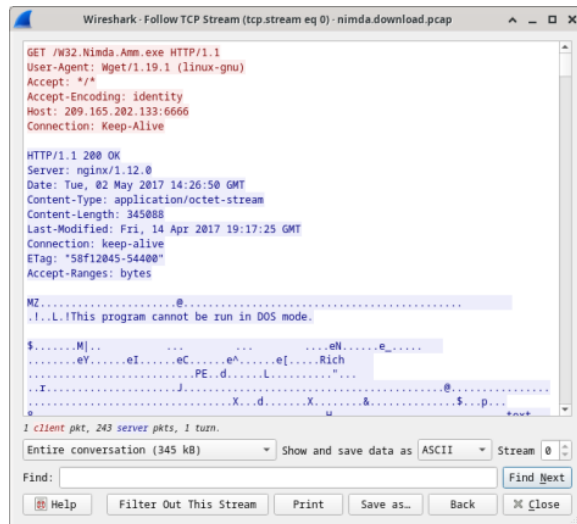


4. Packets one through three are the TCP handshake. The fourth packet shows the request for the malware file. Confirming what was already known, the request was done over HTTP, sent as a GET request.

5. Because HTTP runs over TCP, it is possible to use Wireshark's Follow TCP Stream feature to rebuild the TCP transaction. Select the first TCP packet in the capture, a SYN packet. Right-click it and choose Follow > TCP Stream.



**6.** Wireshark displays another window containing the details for the entire selected TCP flow.



Questions:

**What are all those symbols shown in the Follow TCP Stream window? Are they connection noise? Data? Explain.**

The symbols are the actual contents of the downloaded file. Because it is binary file, Wireshark does not know how to represent it. The displayed symbols are Wireshark's best guess at making sense of the binary data while decoding it as text.

**There are a few readable words spread among the symbols. Why are they there?**

Those are strings contained in the executable code. Usually, these words are part of messages provided by the program to the user while it runs. While more of an art than a science, a skilled analyst can extract valuable information by reading through these fragments.

Challenge Question: Despite the W32.Nimda.Amm.exe name, this executable is not the famous worm. For security reasons, this is another executable file that was renamed as W32.Nimda.Amm.exe. Using the word fragments displayed by Wireshark's Follow TCP Stream window, can you tell what executable this really is?

Scrolling all the way down on that window reveals that this is the Microsoft Windows cmd.exe file.

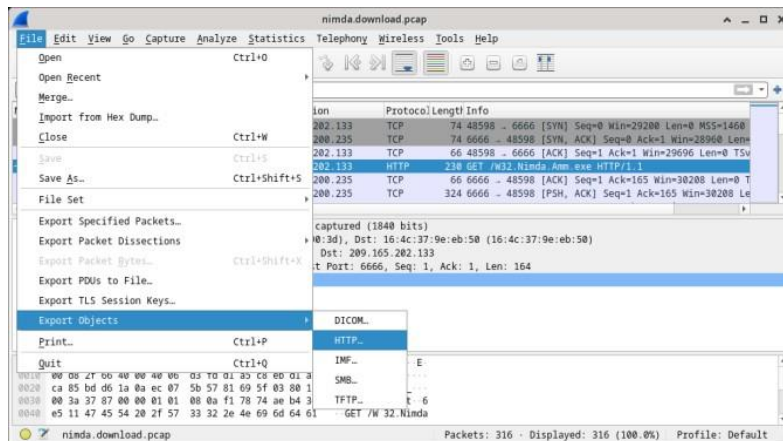
7. Click Close in the Follow TCP Stream window to return to the Wireshark nimda.download.pcap file

## Part 2: Extract Downloaded Files from PCAP

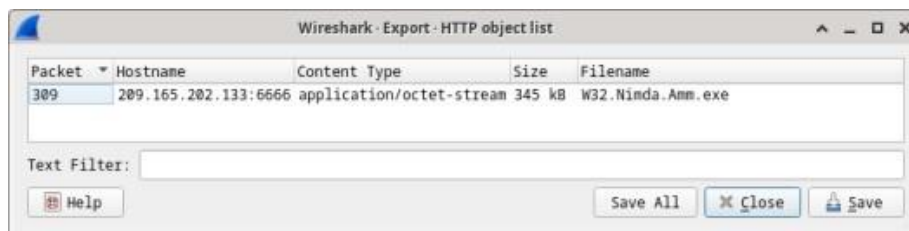
Because capture files contain all packets related to traffic, a PCAP of a download can be used to retrieve a previously downloaded file. Follow the steps below to use Wireshark to retrieve the Nimda malware.

In that fourth packet in the `nimda.download.pcap` file, notice that the HTTP GET request was generated from 209.165.200.235 to 209.165.202.133. The Info column also shows this is in fact the GET request for the file.

With the GET request packet selected, navigate to File > Export Objects > HTTP, from Wireshark's menu.



Wireshark will display all HTTP objects present in the TCP flow that contains the GET request. In this case, only the W32.Nimda.Amm.exe file is present in the capture. It will take a few seconds before the file is displayed.



Question:

**Why is W32.Nimda.Amm.exe the only file in the capture?**

Because the capture was started right before the download and stopped right after. No other traffic was caught while the capture was active.

In the HTTP object list window, select the W32.Nimda.Amm.exe file and click Save As at the bottom of the screen.

Click the left arrow until you see the Home button. Click Home and then click the analyst folder (not the analyst tab). Save the file there.

Return to your terminal window and ensure the file was saved. Change directory to the /home/analyst folder and list the files in the folder using the ls -l command.

```
[analyst@secOps pcaps]$ cd /home/analyst
```

```
[analyst@secOps ~]$ ls -l total 364 drwxr-xr-x 2 analyst
```

```
analyst  4096 Sep 26  2014 Desktop drwxr-xr-x 3 analyst
```

```
analyst  4096 May 25 11:16 Downloads drwxr-xr-x 2
```

```
analyst analyst 4096 May 22 08:39 extra drwxr-xr-x 8
```

```
analyst analyst 4096 Jun 22 11:38 lab.support.files drwxr-  
xr-x 2 analyst analyst 4096 Mar 3 15:56 second_drive  
-rw-r-- 1 analyst analyst 345088 Jun 22 15:12 W32.Nimda.Amm.exe
```

[analyst@secOps ~]\$ Question:

**Was the file saved?**

Yes

The file command gives information on the file type. Use the file command to learn a little more about the malware, as show below:

```
[analyst@secOps ~]$ file W32.Nimda.Amm.exe
```

W32.Nimda.Amm.exe: PE32+ executable (console) x86-64, for MS Windows

```
[analyst@secOps ~]$
```

As seen above, W32.Nimda.Amm.exe is indeed a Windows executable file.

Question:

**In the malware analysis process, what would be a probable next step for a security analyst?**

The goal is to identify the type of malware and analyze its behavior. Therefore, the malware file should be moved to a controlled environment and execute it to watch its behavior. Malware analysis environments often rely on virtual machines and are sandboxed to avoid damage to non-test systems. Such environments usually contain tools that facilitate monitoring of the malware execution; resources usage, network connections and operating system changes are common monitored aspects.

There are also a few Internet-based malware analysis tools. VirusTotal (virustotal.com) is one example. Analysts upload malware to VirusTotal, which in turn, executes the malicious code. After execution and a number of other checks, VirusTotal returns a report to the analyst.

## PRACTICAL 4

### Demonstrate Analysis of DNS Traffic

#### Theory:

Wireshark is an open source packet capture and analysis tool. Wireshark gives a detailed breakdown of the network protocol stack. Wireshark allows you to filter traffic for network troubleshooting, investigate security issues, and analyze network protocols. Because Wireshark allows you to view the packet details, it can be used as a reconnaissance tool for an attacker.

In this lab, you will install Wireshark on a Windows system and use Wireshark to filter for

DNS packets and view the details of both DNS query and response packets **Steps:**

#### Part 1: Capture DNS Traffic

Step 1: Download and install Wireshark.

- a. Download the latest stable version of Wireshark from [www.wireshark.org](http://www.wireshark.org). Choose the software version you need based on your PC's architecture and operating system.
- b. Follow the on-screen instructions to install Wireshark. If you are prompted to install USBPcap, **do NOT** install USBPcap for normal traffic capture. USBPcap is experimental, and it could cause USB problems on your PC.

Step 2: Capture DNS traffic.

- a. Start Wireshark. Select an active interface with traffic for packet capture.
- b. Clear the DNS cache.
  - 1) In Windows, enter **ipconfig /flushdns** in Command Prompt.
  - 2) For the majority of Linux distributions, one of the following utilities is used for DNS caching: Systemd -Resolved, DNSMasq, and NSCD. If your Linux distribution does not use one of the listed utilities, please perform an internet search for the DNS caching utility for your Linux distribution

(i) Identify the utility used in your Linux distribution by checking the status:

Systemd-Resolved: **systemctl status systemd-resolved.service**

DNSMasq: **systemctl status dnsmasq.service**

NSCD: **systemctl status nscd.service**

(ii) If you are using system-resolved, enter **systemd-resolve --flush-caches** to flush the cache for Systemd-Resolved before restarting the service. The following commands restart the associated service using elevated privileges:

Systemd-Resolved: **sudo systemctl restart systemd-resolved.service**

DNSMasq: **sudo systemctl restart dnsmasq.service**

NSCD: **sudo systemctl restart nscd.service**

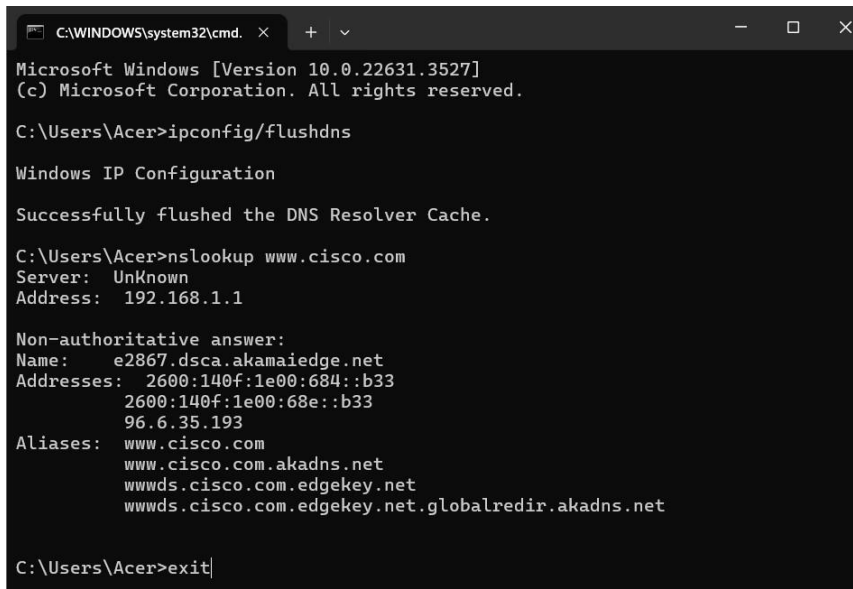


3) For the macOS, enter **sudo killall -HUP mDNSResponder** to clear the DNS cache in the Terminal. Perform an internet search for the commands to clear the DNS cache for an older OS.

c. At a command prompt or terminal, type **nslookup** enter the interactive mode.

d. Enter the domain name of a website. The domain name [www.cisco.com](http://www.cisco.com) is used in this example.

e. Type **exit** when finished. Close the command prompt.



```
C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [Version 10.0.22631.3527]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Acer>ipconfig /flushdns

Windows IP Configuration

Successfully flushed the DNS Resolver Cache.

C:\Users\Acer>nslookup www.cisco.com
Server: UnKnown
Address: 192.168.1.1

Non-authoritative answer:
Name: e2867.dsca.akamaiedge.net
Addresses: 2600:140f:1e00:684::b33
           2600:140f:1e00:68e::b33
           96.6.35.193
Aliases: www.cisco.com
          www.cisco.com.akadns.net
          wwwds.cisco.com.edgekey.net
          wwwds.cisco.com.edgekey.net.globalredir.akadns.net

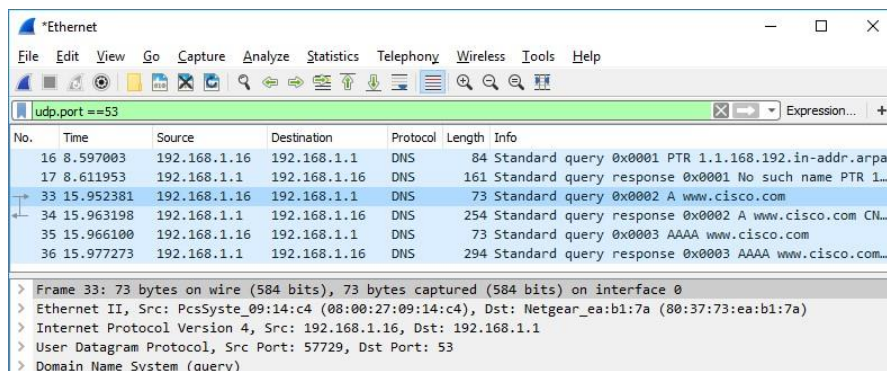
C:\Users\Acer>exit
```

f. Click **Stop capturing packets** to stop the Wireshark capture.

## Part 2: Explore DNS Query Traffic

a. Observe the traffic captured in the Wireshark Packet List pane. Enter **udp.port == 53** in the filter box and click the arrow (or press enter) to display only DNS packets.

**Note:** The provided screenshots are just examples. Your output maybe slightly different.

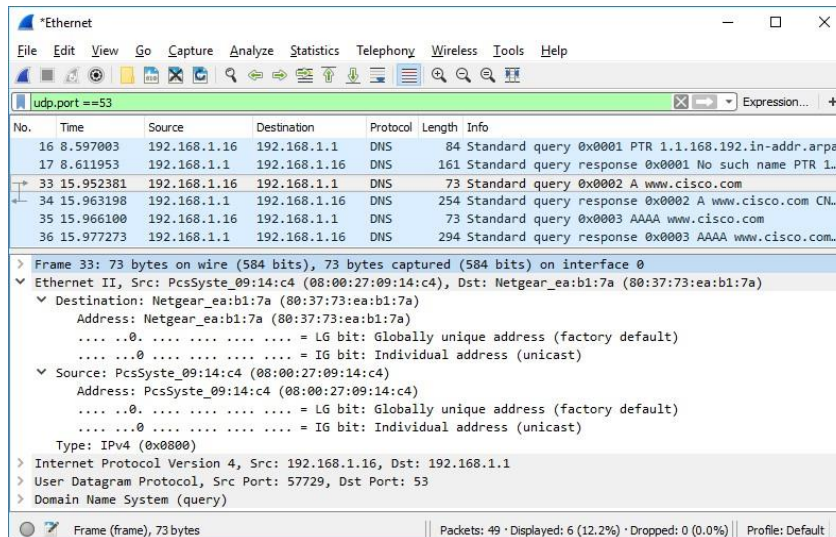


b. Select the DNS packet contains **Standard query** and **A www.cisco.com** in the Info column.

c. In the Packet Details pane, notice this packet has Ethernet II, Internet Protocol Version 4, User Datagram Protocol and Domain Name System (query).

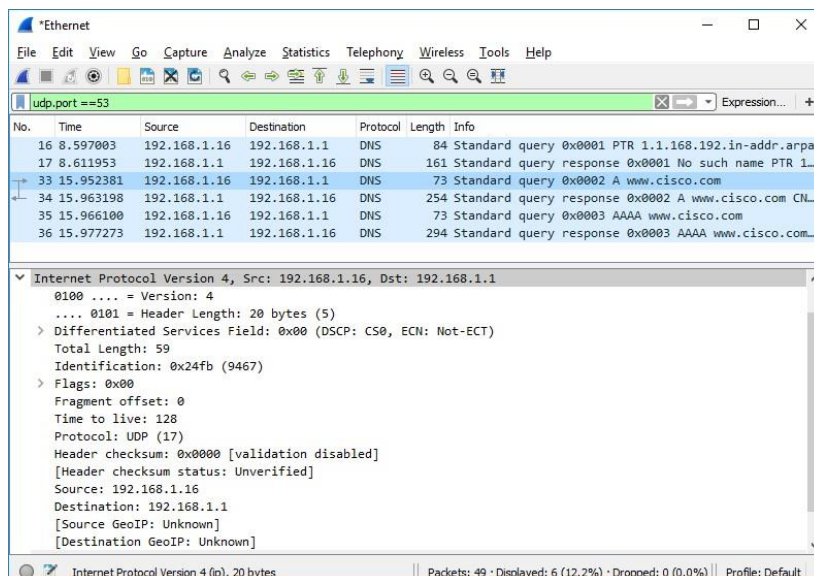


d. Expand **Ethernet II** to view the details. Observe the source and destination fields.



What are the source and destination MAC addresses? Which network interfaces are these MAC addresses associated with?

In this example, the source MAC address is associated with the NIC on the PC and the destination MAC address is associated with the default gateway. If there is a local DNS server, the destination MAC address would be the MAC address of the local DNS server. e. Expand **Internet Protocol Version 4**. Observe the source and destination IPv4 addresses.



What are the source and destination IP addresses? Which network interfaces are these IP addresses associated with?

In this example, the source IP address is associated with the NIC on the PC and the destination IP address is associated with the default gateway.

f. Expand the **User Datagram Protocol**. Observe the source and destination ports.

The image shows a Wireshark packet capture window titled '\*Ethernet'. The filter bar at the top shows 'udp.port == 53'. The packet list table below shows several DNS packets. Packet 33 is selected, and its details are expanded in the packet details pane.

No.	Time	Source	Destination	Protocol	Length	Info
16	8.597003	192.168.1.16	192.168.1.1	DNS	84	Standard query 0x0001 PTR 1.1.168.192.in-addr.arpa
17	8.611953	192.168.1.1	192.168.1.16	DNS	161	Standard query response 0x0001 No such name PTR 1...
33	15.952381	192.168.1.16	192.168.1.1	DNS	73	Standard query 0x0002 A www.cisco.com
34	15.963198	192.168.1.1	192.168.1.16	DNS	254	Standard query response 0x0002 A www.cisco.com CN...
35	15.966100	192.168.1.16	192.168.1.1	DNS	73	Standard query 0x0003 AAAA www.cisco.com
36	15.977273	192.168.1.1	192.168.1.16	DNS	294	Standard query response 0x0003 AAAA www.cisco.com...

Details of Frame 33:

- Frame 33: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface 0
- Ethernet II, Src: PcsSyste\_09:14:c4 (08:00:27:09:14:c4), Dst: Netgear\_ea:b1:7a (80:37:73:ea:b1:7a)
- Internet Protocol Version 4, Src: 192.168.1.16, Dst: 192.168.1.1
- User Datagram Protocol, Src Port: 57729, Dst Port: 53
  - Source Port: 57729
  - Destination Port: 53
  - Length: 39
  - Checksum: 0x839a [unverified]
  - [Checksum Status: Unverified]
  - [Stream index: 2]
- Domain Name System (query)

Summary: User Datagram Protocol (udp), 8 bytes | Packets: 49 · Displayed: 6 (12.2%) · Dropped: 0 (0.0%) | Profile: Default

What are the source and destination ports? What is the default DNS port number?

The source port number is 577729 and the destination port is 53, which is the default DNS port number.

g. Determine the IP and MAC address of the PC.

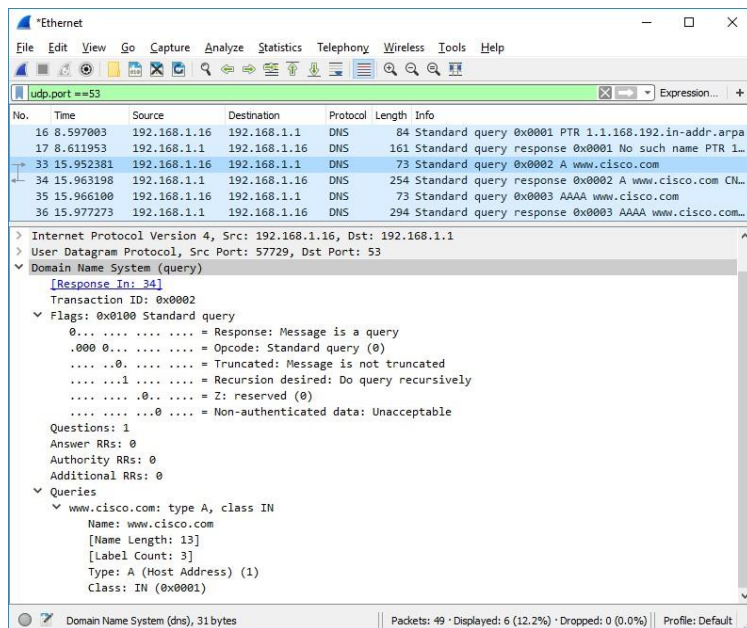
1. In a Windows command prompt, enter **arp -a** and **ipconfig /all** to record the MAC and IP addresses of the PC.
2. For Linux and macOS PC, enter **ifconfig** or **ip address** in a terminal.

Compare the MAC and IP addresses in the Wireshark results to the IP and MAC addresses. What is your observation?

The IP and MAC addresses captured in the Wireshark results are the same as the addresses listed in ipconfig /all command.

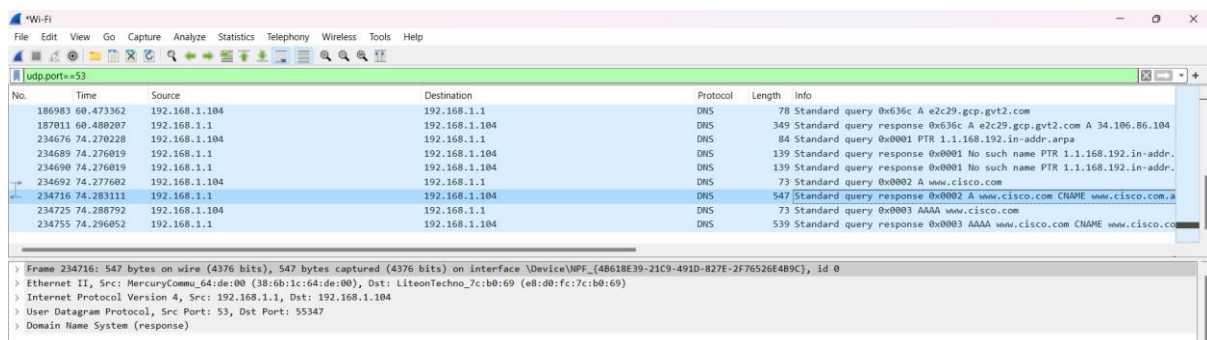
h. Expand **Domain Name System (query)** in the Packet Details pane. Then expand the **Flags** and **Queries**.

i. Observe the results. The flag is set to do the query recursively to query for the IP address to www.cisco.com.



## Part 3: Explore DNS Response Traffic

- a. Select the corresponding response DNS packet has **Standard query response** and **A** **www.cisco.com** in the Info column.



What are the source and destination MAC and IP addresses and port numbers? How do they compare to the addresses in the DNS query packets?

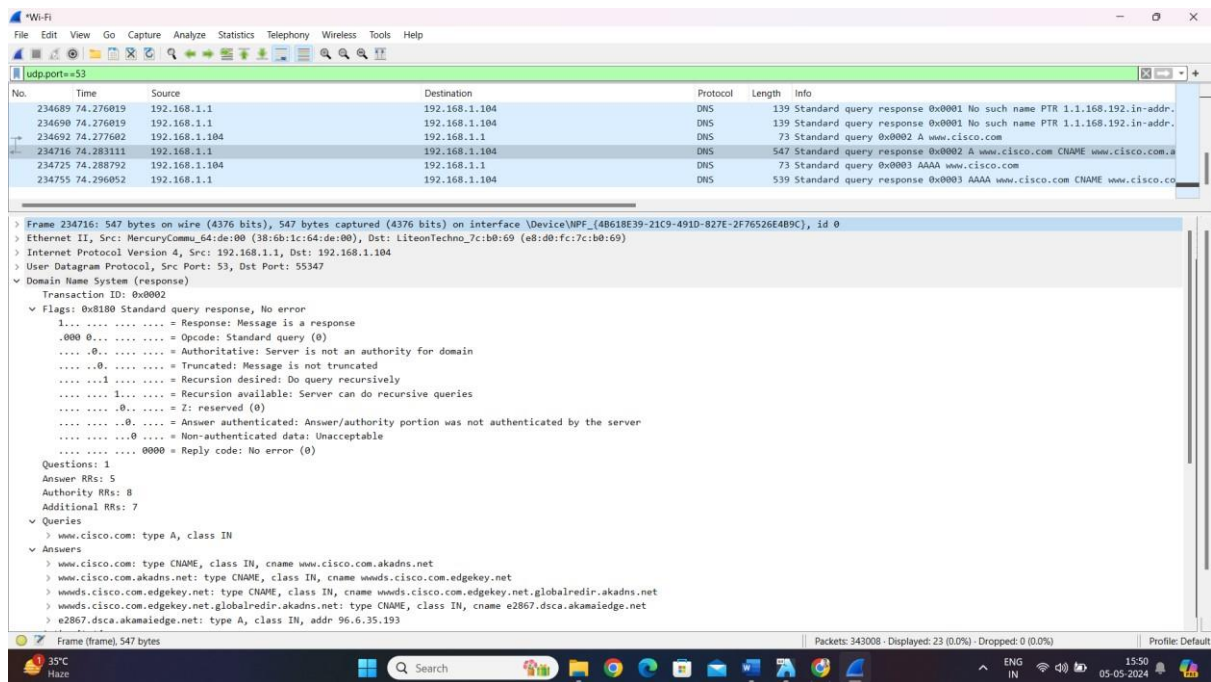
The source IP, MAC address, and port number in the query packet are now destination addresses. The destination IP, MAC address, and port number in the query packet are now source addresses.

- b. Expand **Domain Name System (response)**. Then expand the **Flags**, **Queries**, and **Answers**.

- c. Observe the results.

Can the DNS server do recursive queries?

Yes, the DNS can handle recursive queries.



d. Observe the CNAME and A records in the Answers details.

How do the results compare to nslookup results?

The results in the Wireshark should be the same as the results from nslookup in the Command Prompt or terminal.

## Reflection

1. From the Wireshark results, what else can you learn about the network when you remove the filter?

Without the filters, the results display other packets, such as DHCP and ARP. From these packets and the information contained within these packets, you can learn about other devices and their functions within the LAN.

2. How can an attacker use Wireshark to compromise your network security?

An attacker on the LAN can use Wireshark to observe the network traffic and can get sensitive information in the packet details if the traffic is not encrypted.

## PRACTICAL 5

### Create your own syslog Server using Ubuntu

Theory:

[Rsyslog](#) is a rocket-fast system for log processing and is commonly used for any kind of system logging. We use a Ubuntu server 20.04 LTS distribution to show you how to configure your own syslog server to receive your CDN logs in real time. **Steps:**

Syslog server installation#

Update the packages list and install the latest version of rsyslog.

1. apt update
2. apt install rsyslog

Syslog server configuration#

Configure rsyslog to receive UDP logs and define a filter where you want to store the logs.

1. Open the rsyslog.conf file and add the following lines.

```
vi /etc/rsyslog.conf
```

```
# provides UDP syslog reception
module(load="imudp")
```

2. Create and open your custom config file. vi

```
/etc/rsyslog.d/00-custom.conf
```

```
# Templates template(name="ReceiveFormat" type="string"
string="%msg:39:$%\n")
```

```
# UDP ruleset mapping
input(type="imudp" port="514" ruleset="customRuleset")
```

```
# Custom ruleset
ruleset(name="customRuleset") {  if ($msg contains '366c3df6-
93dd-4ec0-a218-aec9d191c59e') then {
    /var/log/cdn.log;ReceiveFormat
stop
```



```
}  
}
```

Replace 366c3df6-93dd-4ec0-a218-aec9d191c59e with your own custom token. Your token values must be between 8 to 45 characters.

Use the following regex expression with [regex101](#) to validate the token value you define:

```
^[a-zA-Z0-9\-\_]*$
```

You may use any letters from a-zA-Z, - and numbers from 0-9 when creating your token.

3. Restart the rsyslog process.

```
systemctl restart rsyslog
```

4. Configure [Log Forwarding](#) in the KeyCDN dashboard with your syslog server details.
5. Verify if you are receiving the logs (log forwarding starts within 5 minutes). `tail -f`

```
/var/log/cdn.log
```

## Troubleshooting commands

- `systemctl status rsyslog` Verify that rsyslog is running.
- `netstat -na | grep "<defined port>"` Is rsyslog listening on the right port? #

```
netstat -na | grep :514 udp      0      0 0.0.0.0:514      0.0.0.0:*
```

- `tcpdump port <defined port>` Are you receiving any packet on the defined port?

```
# tcpdump port 514 tcpdump: verbose output suppressed, use -v or -vv for full
```

```
protocol decode listening on eth0, link-type EN10MB (Ethernet), capture size 65535
```

```
bytes
```

```
11:20:53.066938 IP keycdn-syslog.37960 > your-server.syslog: [syslog]
```

```
^C
```

```
1 packet captured
```

```
1 packet received by filter
```

```
0 packets dropped by kernel
```

- `tail -f /path/to/your/logfile` Check if you get new log entries.

```
# tail -f /var/log/cdn.log
```

1421338853.058|defr|115.81.56.12|200|439|1|6976|zonename-hexid.kxcdn.com|HIT|"HEAD  
/lorem.jpg HTTP/1.1"|15/Jan/2015:17:20:53  
+0100|""|"curl/7.30.0"|http|CH|Switzerland|Winterthur|25|47.5000|8.7251|"AS6830 Liberty  
Global Operations B.V."^C

## Create your own syslog Server using Kali

### Step 1 : Installation

```
root@kali:~# systemctl status rsyslog.service
● rsyslog.service - System Logging Service
   Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled; vendor preset: en
   Active: active (running) since Fri 2024-05-03 10:47:36 EDT; 1 day 19h ago
   TriggeredBy: ● syslog.socket
     Docs: man:rsyslogd(8)
           https://www.rsyslog.com/doc/
    Main PID: 21259 (rsyslogd)
      Tasks: 4 (limit: 2337)
     Memory: 1.1M
    CGroup: /system.slice/rsyslog.service
            └─21259 /usr/sbin/rsyslogd -n -iNONE

May 03 10:47:36 kali systemd[1]: Starting System Logging Service...
May 03 10:47:36 kali rsyslogd[21259]: imuxsock: Acquired UNIX socket '/run/systemd/j
May 03 10:47:36 kali rsyslogd[21259]: [origin software="rsyslogd" swVersion="8.2001.
May 03 10:47:36 kali systemd[1]: Started System Logging Service.
May 05 06:08:55 kali rsyslogd[21259]: [origin software="rsyslogd" swVersion="8.2001.
May 05 06:08:55 kali rsyslogd[21259]: [origin software="rsyslogd" swVersion="8.2001.

root@kali:~# systemctl start rsyslog.service
root@kali:~# systemctl start rsyslog.service -l
root@kali:~#
```

### Step 2: Configuration

```
root@kali:~# systemctl start rsyslog.service
root@kali:~# systemctl start rsyslog.service -l
root@kali:~# cat /etc/rsyslog.conf
# /etc/rsyslog.conf configuration file for rsyslog
#
# For more information install rsyslog-doc and see
# /usr/share/doc/rsyslog-doc/html/configuration/index.html

#####
### MODULES ###
#####

module(load="imuxsock") # provides support for local system logging
module(load="imklog")   # provides kernel logging support
#module(load="immark")  # provides --MARK-- message capability
```

### Step 3: Test Syslog with CLI

```
root@kali:~# ls /etc/rsyslog.d/
root@kali:~# logger -h

Usage:
  logger [options] [<message>]

Enter messages into the system log.

Options:
  -i                log the logger command's PID
  -id[=<id>]        log the given <id>, or otherwise the PID
  -f, --file <file> log the contents of this file
  -e, --skip-empty   do not log empty lines when processing files
  -no-act           do everything except the write the log
  -p, --priority <prio> mark given message with this priority
  --octet-count     use rfc6587 octet counting
  --prio-prefix     look for a prefix on every line read from stdin
  -s, --stderr      output message to standard error as well
```

```
For more details see logger(1).
root@kali:~# logger "Hi This is my message"
root@kali:~# cat /var/log/messages | grep -i "Hi This is my message"
May  5 06:24:13 kali root: Hi This is my message
root@kali:~#
```

```
root@kali:~# logger -p emerg "Hi This is my message"
root@kali:~#
root@kali:~# logger -t myapp -p emerg "this is my message"
root@kali:~# logger -t my_App "Hi, This is a test message"
root@kali:~# cat /var/log/messages | grep -i my_App
May  5 06:30:14 kali my_App: Hi, This is a test message
root@kali:~#
```

## PRACTICAL 6

### Configure your Linux system to send syslog messages to a syslog server and Read them Steps:

To configure your Linux system to send syslog messages to a syslog server and read them, you'll need to follow these general steps:

#### 1. Install and configure a syslog server:

```
root@kali:~# systemctl status rsyslog.service
● rsyslog.service - System Logging Service
   Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled; vendor preset: en
   Active: active (running) since Fri 2024-05-03 10:47:36 EDT; 1 day 19h ago
   TriggeredBy: ● syslog.socket
     Docs: man:rsyslogd(8)
           https://www.rsyslog.com/doc/
    Main PID: 21259 (rsyslogd)
      Tasks: 4 (limit: 2337)
     Memory: 1.1M
    CGroup: /system.slice/rsyslog.service
            └─21259 /usr/sbin/rsyslogd -n -iNONE

May 03 10:47:36 kali systemd[1]: Starting System Logging Service ...
May 03 10:47:36 kali rsyslogd[21259]: imuxsock: Acquired UNIX socket '/run/systemd/j>
May 03 10:47:36 kali rsyslogd[21259]: [origin software="rsyslogd" swVersion="8.2001.>
May 03 10:47:36 kali systemd[1]: Started System Logging Service.
May 05 06:08:55 kali rsyslogd[21259]: [origin software="rsyslogd" swVersion="8.2001.>
May 05 06:08:55 kali rsyslogd[21259]: [origin software="rsyslogd" swVersion="8.2001.>

root@kali:~# systemctl start rsyslog.service
root@kali:~# systemctl start rsyslog.service -l
root@kali:~#
```

```
root@kali:~# systemctl start rsyslog.service
root@kali:~# systemctl start rsyslog.service -l
root@kali:~# cat /etc/rsyslog.conf
# /etc/rsyslog.conf configuration file for rsyslog
#
# For more information install rsyslog-doc and see
# /usr/share/doc/rsyslog-doc/html/configuration/index.html

#####
### MODULES ###
#####

module(load="imuxsock") # provides support for local system logging
module(load="imklog")   # provides kernel logging support
#module(load="immark")  # provides --MARK-- message capability
```

Choose a syslog server software such as rsyslog, syslog-ng, or syslogd. Install the server software on a separate machine or a dedicated server. Configure the syslog server to listen for incoming syslog messages.

#### 2. Configure the client system:

On the Linux system from which you want to send syslog messages, you need to modify the syslog configuration file.

- Locate and open the syslog configuration file. It is typically located at `/etc/rsyslog.conf` or `/etc/syslog-ng/syslog-ng.conf`, depending on the syslog software you are using.
- Look for the line that begins with `*.*` or `*.info`. This line specifies the log level and destination for syslog messages.



- Modify the line to include the IP address or hostname of your syslog server. For example, if your syslog server's IP address is **192.168.0.10**, change the line to **\*.\*@192.168.0.10:514** to send all log messages to that IP address on port 514.
- Save the changes and exit the configuration file.

### 3. Restart syslog service:

After modifying the syslog configuration file, restart the syslog service on the client system to apply the changes. The command to restart the service depends on the Linux distribution you're using. Here are a few common commands:

- For systems using rsyslog: ``sudo service rsyslog restart`` or ``sudo systemctl restart rsyslog``
- For systems using syslog-ng: ``sudo service syslog-ng restart`` or ``sudo systemctl restart syslog-ng``
- For systems using syslogd: ``sudo service syslog restart`` or ``sudo systemctl restart syslog``

### 4. Verify syslog messages are being sent:

To ensure that syslog messages are being sent to the syslog server, you can check the logs on the server side or use a network traffic monitoring tool like Wireshark to monitor network traffic on the syslog server's port (514 by default).

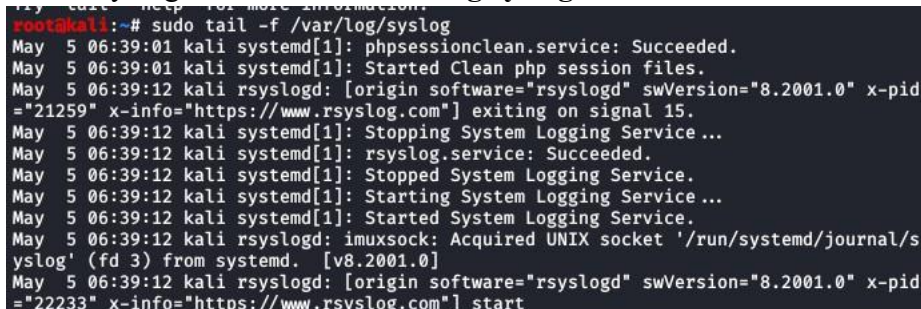
### 5. Read syslog messages on the syslog server:

Once the syslog server is receiving messages from the client, you can read and analyze them.

- Log in to the syslog server machine.
- Locate the logs directory or file, which varies depending on the syslog server software and configuration.

Use a log viewer or a command-line tool to read the syslog messages. For example:

- For rsyslog: ``sudo tail -f /var/log/syslog``
- For syslog-ng: ``sudo tail -f /var/log/messages``
- For syslogd: ``sudo tail -f /var/log/syslog``



```
root@kali:~# sudo tail -f /var/log/syslog
May  5 06:39:01 kali systemd[1]: phpsessionclean.service: Succeeded.
May  5 06:39:01 kali systemd[1]: Started Clean php session files.
May  5 06:39:12 kali rsyslogd: [origin software="rsyslogd" swVersion="8.2001.0" x-pid="21259" x-info="https://www.rsyslog.com"] exiting on signal 15.
May  5 06:39:12 kali systemd[1]: Stopping System Logging Service ...
May  5 06:39:12 kali systemd[1]: rsyslog.service: Succeeded.
May  5 06:39:12 kali systemd[1]: Stopped System Logging Service.
May  5 06:39:12 kali systemd[1]: Starting System Logging Service ...
May  5 06:39:12 kali systemd[1]: Started System Logging Service.
May  5 06:39:12 kali rsyslogd: imuxsock: Acquired UNIX socket '/run/systemd/journal/syslog' (fd 3) from systemd. [v8.2001.0]
May  5 06:39:12 kali rsyslogd: [origin software="rsyslogd" swVersion="8.2001.0" x-pid="22233" x-info="https://www.rsyslog.com"] start
```

Adjust the log file path according to your specific syslog server configuration.

By following these steps, you should be able to configure your Linux system to send syslog messages to a syslog server and read them on the server side.

## PRACTICAL 7

### Install and Run Splunk on Linux

#### Theory:

Splunk Enterprise is a SIEM application that gathers, organizes, and visualizes machinegenerated log data from local and remote machines, websites, and cloud services. Having Splunk setup with your physical and cloud systems can be valuable for staying ahead of cybersecurity, connectivity, and other big data initiatives.

#### Steps:

##### Install Splunk Enterprise on Debian/Ubuntu

1. Create an account on on Splunk.com.
2. Select Free Splunk in the upper-right corner.
3. Select Free Splunk.
4. Select Linux, then Download Now beside .deb.
5. Upload the file to your server with SCP, replacing the filename, username, and server hostname as needed:

```
scp splunk-versionnumber.deb root@11.22.33.44:/root
```

6. SSH into your server as root.
7. Install the Splunk Enterprise DEB file: `dpkg -i splunk-file.deb`
8. Verify Splunk installation status:

```
dpkg --status splunk
```

[Continue to Splunk setup.](#)

##### Change Your Default Shell

Splunk recommends using **bash** as your default shell as Debian's default shell, **dash**, may cause zombie processes which cannot be killed. Below we'll cover how to change your default Debian shell. 1)Find your default shell: `which sh`

2) You should see `/bin/sh` or another symbolic link. Use `ls` to find the actual shell: `ls`

```
-l /bin/sh
```

3) If it doesn't show `bash` at the end, view installed shells to ensure it is installed:

```
cat /etc/shells
```

4) Delete the symbolic link: `rm /bin/sh`

5) Create a new symbolic link pointing /bin/sh to bash:

```
ln -s /bin/bash /bin/sh
```

### Complete Splunk Setup:

After you install Splunk, follow the steps below to complete your Splunk setup.

1) Use Splunk to start the Splunk service:

```
/opt/splunk/bin/splunk start
```

2) Read the license agreement. At the end, select **y** and **Enter**.

3) Create an username.

4) Create a password with at least eight characters.

5) Once Splunk installation is complete, the last line will provide the URL to access the web interface: `http://serverhostname:8000`.

6) Open port 8000 in your firewall: FirewallD, UFW, CSF, etc.

Keep in mind that if you get locked out of your server and restart it, you'll need to start the Splunk service again before you can access the Splunk dashboard.

Open your Splunk web interface in your browser.

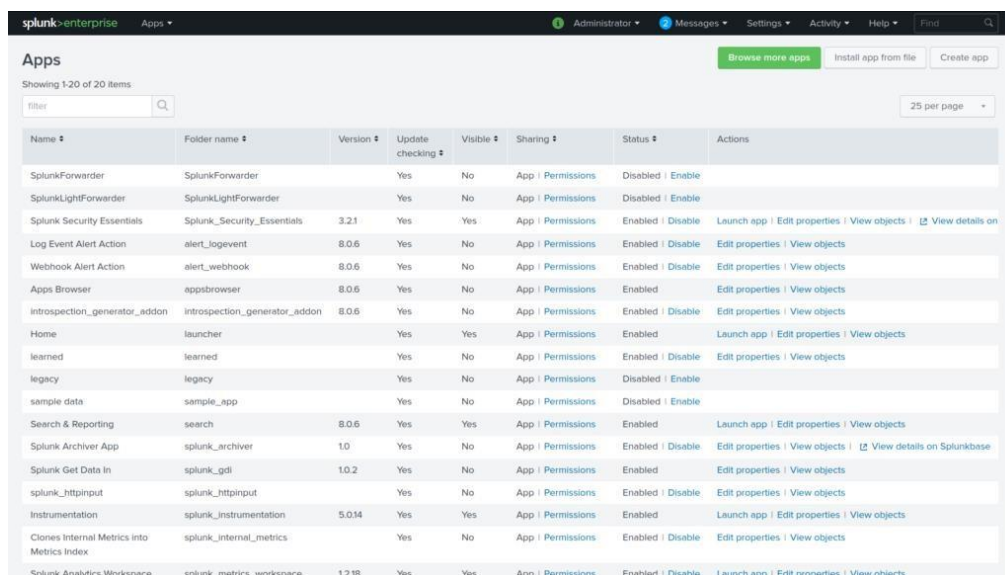
### Monitor Data in Splunk:

1. Log into your Splunk web interface.
2. Select Add Data.
3. At the bottom, select Monitor.
4. On the left of the Select Source page, select Files & Directories.
5. Select Browse.
6. Specify a file or directory to monitor and click Select. For example, you can monitor a cPanel log, Apache access log (similar to GoAccess Analytics), or a cPanel user directory. We'll use the `/var/log/secure` log file which tracks SSH logins, and authentication failures on CentOS. Debian/Ubuntu users will instead use `/var/log/auth.log`.
7. Select Continuously Monitor to show updates to the log file in real-time.
8. At the top, select Next to access the Set Source Type page.
9. Source type on the left should state "linux\_secure" so Splunk knows it is Linux security information. Otherwise, select the button and select linux\_secure from the drop-down menu.
10. Select Next at the top.

11. (Optional) On the Input Settings page, you can change the App context, machine hostname, and Index.
12. Select Review.
13. Ensure everything is correct and select Submit.
14. You'll see "File input has been created successfully." Start Searching.
15. To modify the event results, select Settings and Data Inputs from the upper-right corner.
16. Select the data input type on the left. For this example, we'll select Files & Directories.
17. Select Enable or Disable for the data path in the Status column.

From your Splunk dashboard, select the gear icon on the left beside Apps. On other pages, select Apps and Manage Apps from the top-left of the page. From here you can:

- Disable and enable installed apps
- Modify permissions
- Update settings
- Launch apps



Name	Folder name	Version	Update checking	Visible	Sharing	Status	Actions
SplunkForwarder	SplunkForwarder		Yes	No	App   Permissions	Disabled   Enable	
SplunkLightForwarder	SplunkLightForwarder		Yes	No	App   Permissions	Disabled   Enable	
Splunk Security Essentials	Splunk_Security_Essentials	3.2.1	Yes	Yes	App   Permissions	Enabled   Disable	Launch app   Edit properties   View objects   View details on
Log Event Alert Action	alert_logevent	8.0.6	Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
Webhook Alert Action	alert_webhook	8.0.6	Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
Apps Browser	appsbrowser	8.0.6	Yes	No	App   Permissions	Enabled	Edit properties   View objects
Introspection_generator_addon	introspection_generator_addon	8.0.6	Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
Home	launcher		Yes	Yes	App   Permissions	Enabled	Launch app   Edit properties   View objects
learned	learned		Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
legacy	legacy		Yes	No	App   Permissions	Disabled   Enable	
sample data	sample_app		Yes	No	App   Permissions	Disabled   Enable	
Search & Reporting	search	8.0.6	Yes	Yes	App   Permissions	Enabled	Launch app   Edit properties   View objects
Splunk Archiver App	splunk_archiver	1.0	Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects   View details on Splunkbase
Splunk Get Data In	splunk_gdi	1.0.2	Yes	No	App   Permissions	Enabled	Edit properties   View objects
splunk_httpinput	splunk_httpinput		Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
Instrumentation	splunk_instrumentation	5.0.14	Yes	Yes	App   Permissions	Enabled	Launch app   Edit properties   View objects
Clones Internal Metrics into Metrics Index	splunk_internal_metrics		Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
Splunk Analytics Workspace	splunk_metrics_workspace	1.2.18	Yes	Yes	App   Permissions	Enabled   Disable	Launch app   Edit properties   View objects

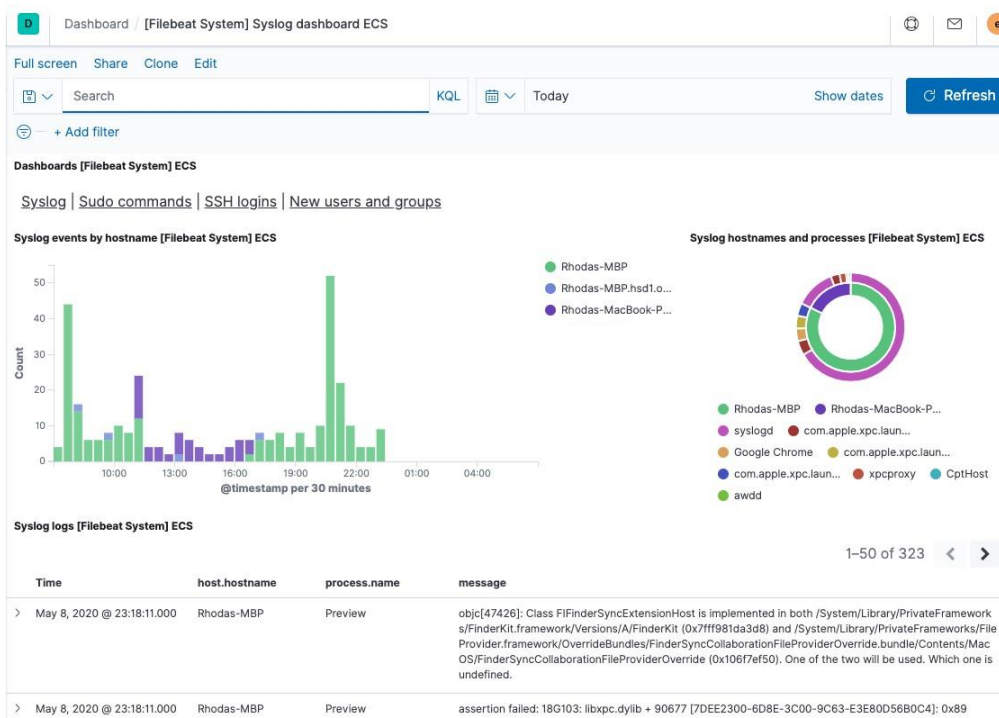
## PRACTICAL 8

### Install and Configure ELK on Linux

#### Filebeat quick start: installation and configuration

This guide describes how to get started quickly with log collection. You'll learn how to:

- install Filebeat on each system you want to monitor
- specify the location of your log files
- parse log data into fields and send it to Elasticsearch
- visualize the log data in Kibana



Before you begin

You need Elasticsearch for storing and searching your data, and Kibana for visualizing and managing it.

Elasticsearch Service Self-managed

To get started quickly, spin up a deployment of our [hosted Elasticsearch Service](#). The Elasticsearch Service is available on AWS, GCP, and Azure. [Try it out for free](#).

## Step 1: Install Filebeat

Install Filebeat on all the servers you want to monitor.

To download and install Filebeat, use the commands that work with your system:

```
curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.8.1-linux-x86\_64.tar.gz
```

```
tar xzvf filebeat-8.8.1-linux-x86_64.tar.gz
```

### Other installation options

- APT or YUM
- Download page
- Docker
- Kubernetes
- Cloud Foundry

## Step 2: Connect to the Elastic Stack

Connections to Elasticsearch and Kibana are required to set up Filebeat.

Set the connection information in filebeat.yml. To locate this configuration file, see Directory layout.

### Elasticsearch Service Self-managed

Specify the cloud.id of your Elasticsearch Service, and set cloud.auth to a user who is authorized to set up Filebeat. For example:

```
cloud.id:  
"staging:dXMtZWZkdC0xLmF3cy5mb3VuZC5pbyRjZWZWM2ZjI2MWE3NGJmMjRj  
ZTMzYmI4ODExYjg0Mjk0ZiRjNmMyY2E2ZDA0MjI0WFmMGNjN2Q3YTllOT  
YyNTc0Mw=="
```

```
cloud.auth: "filebeat_setup:YOUR_PASSWORD"
```

This example shows a hard-coded password, but you should store sensitive values in the secrets keystore

To learn more about required roles and privileges, see Grant users access to secured resources.

Step 3: Collect log data.

There are several ways to collect log data with Filebeat:

- Data collection modules — simplify the collection, parsing, and visualization of common log formats
- ECS loggers — structure and format application logs into ECS-compatible JSON
- Manual Filebeat configuration

### Enable and configure data collection modules[edit](#)

1. Identify the modules you need to enable. To see a list of available [modules](#), run:

DEB RPM MacOS Linux Windows

`./filebeat modules list`

2. From the installation directory, enable one or more modules. For example, the following command enables the `nginx` module config:

DEB RPM MacOS Linux Windows

`./filebeat modules enable nginx`

3. In the module config under `modules.d`, change the module settings to match your environment. You must enable at least one fileset in the module. **Filesets are disabled by default.**

For example, log locations are set based on the OS. If your logs aren't in default locations, set the `paths` variable:

```
- module: nginx
  access:
    enabled: true
  var.paths: ["/var/log/nginx/access.log*"]
```

To see the full list of variables for a module, see the documentation under [Modules](#).

To test your configuration file, change to the directory where the Filebeat binary is installed, and run Filebeat in the foreground with the following options specified: `./filebeat test config e`. Make sure your config files are in the path expected by Filebeat (see [Directory layout](#)), or use the `-c` flag to specify the path to the config file.

For more information about configuring Filebeat, also see:

- [Configure Filebeat](#)

- [Config file format](#)
- [filebeat.reference.yml](#): This reference configuration file shows all non-deprecated options. You'll find it in the same location as [filebeat.yml](#).

### Enable and configure ECS loggers for application log collection [edit](#)

While Filebeat can be used to ingest raw, plain-text application logs, we recommend structuring your logs at ingest time. This lets you extract fields, like log level and exception stack traces.

Elastic simplifies this process by providing application log formatters in a variety of popular programming languages. These plugins format your logs into ECS-compatible JSON, which removes the need to manually parse logs.

See [ECS loggers](#) to get started.

### Configure Filebeat manually

If you're unable to find a module for your file type, or can't change your application's log output, see [configure the input](#) manually.

Step 4: Set up assets.

Filebeat comes with predefined assets for parsing, indexing, and visualizing your data. To load these assets:

1. Make sure the user specified in [filebeat.yml](#) is [authorized to set up Filebeat](#).
2. From the installation directory, run:

DEB RPM MacOS Linux Windows

`./filebeat setup -e`

`-e` is optional and sends output to standard error instead of the configured log output.

This step loads the recommended [index template](#) for writing to Elasticsearch and deploys the sample dashboards for visualizing the data in Kibana.

This step does not load the ingest pipelines used to parse log lines. By default, ingest pipelines are set up automatically the first time you run the module and connect to Elasticsearch.

A connection to Elasticsearch (or Elasticsearch Service) is required to set up the initial environment. If you're using a different output, such as Logstash, see:

- [Load the index template manually](#)
- [Load Kibana dashboards](#)
- [Load ingest pipelines](#)

Filebeat should not be used to ingest its own log as this may lead to an infinite loop.



### Step 5: Start Filebeat[edit](#)

Before starting Filebeat, modify the user credentials in `filebeat.yml` and specify a user who is [authorized to publish events](#).

To start Filebeat, run:

```
sudo chown root filebeat.yml
sudo chown root modules.d/nginx.yml
sudo ./filebeat -e
```

You'll be running Filebeat as root, so you need to change ownership of the configuration file and any configurations enabled in the `modules.d` directory, or run Filebeat with `--strict.perms=false` specified. See [Config File Permissions](#).

Filebeat should begin streaming events to Elasticsearch.

### Step 6: View your data in Kibana[edit](#)

Filebeat comes with pre-built Kibana dashboards and UIs for visualizing log data. You loaded the dashboards earlier when you ran the setup command.

To open the dashboards:

1. Launch Kibana:
  - Elasticsearch Service Self-managed
  - a. Log in to your Elastic Cloud account.
  - b. Navigate to the Kibana endpoint in your deployment.
2. In the side navigation, click Discover. To see Filebeat data, make sure the predefined `filebeat-*` index pattern is selected.  
If you don't see data in Kibana, try changing the time filter to a larger range. By default, Kibana shows the last 15 minutes.
3. In the side navigation, click Dashboard, then select the dashboard that you want to open.

The dashboards are provided as examples. We recommend that you customize them to meet your needs.

## PRACTICAL 9

### Install and Configure GrayLog on Linux

#### Prerequisites:

Hint: This guide assumes that any firewall is disabled and traffic can flow across all necessary ports.

Graylog 5.0 requires the following to maintain compatibility with its software dependencies:

- OpenJDK 17 (This is embedded in Graylog 5.0 and does not need to be separately installed.)
- OpenSearch 1.x, 2.x or Elasticsearch 7.10.2
- MongoDB 5.x or 6.x
- 

#### Server Timezone

To set a specific time zone on the Graylog server, you can use the following command. (For more information on setting a time zone, we recommend [this blog post](#).) **sudo timedatectl set-timezone UTC**

#### MongoDB

Graylog 5.0 is compatible with MongoDB 5.x-6.x.

To install MongoDB on Ubuntu, the official MongoDB documentation [provides a helpful tutorial](#).

MongoDB recommends you disable Transparent Huge Pages [Disable Transparent Huge Pages \(THP\)](#)

1. First select your version of Ubuntu and begin the installation sequence:

~~—Ubuntu 20.04~~

~~—Ubuntu 22.04~~

2. You can use a keyserver approach via a widget if you wish to incorporate proxies and other non-free environments. For example:

**wget -qO-**

**'http://keyserver.ubuntu.com/pks/lookup?op=get&search=0xf5679a222c647c87527c2f8c  
b00a0bd1e2c63c11' | sudo apt-key add -**

3. Enable MongoDB during the operating system's start up and verify it is running: **sudo**

**systemctl daemon-reload sudo systemctl enable mongod.service sudo systemctl  
restart mongod.service**

**sudo systemctl --type=service --state=active | grep mongod**

Hint: For the following sections on OpenSearch and Elasticsearch, select which data node you will be using for your Graylog instance and complete only the requisite section.

## OpenSearch

If you are using OpenSearch as your data node, then follow the steps below to install OpenSearch 2.5.0.

The recommended method of installation is to [follow the user documentation](#) provided by the OpenSearch service. To set up the OpenSearch service with your Graylog instance, follow the steps below:

The following example will demonstrate installing OpenSearch using the DEB package.

1. Import the public GPG key. This key is used to verify that the APT repository is signed.

```
curl -o- https://artifacts.opensearch.org/publickeys/opensearch.pgp | sudo apt-key add -
```

2. Create an APT repository for OpenSearch:

```
echo "deb https://artifacts.opensearch.org/releases/bundle/opensearch/2.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/opensearch-2.x.list
```

3. Verify that the repository was created successfully.

```
sudo apt-get update
```

4. With the repository information added, list all available versions of OpenSearch:

```
sudo apt list -a opensearch
```

5. Choose the version of OpenSearch you want to install. (Unless otherwise indicated, the latest available version of OpenSearch is installed.) **sudo apt-get install opensearch**

To install a specific version of OpenSearch, specify the version manually using `opensearch=<version>`. **sudo apt-get install opensearch=2.5.0**

## Graylog Configuration for OpenSearch

1. Begin by opening the yml file: **sudo nano**

```
/etc/opensearch/opensearch.yml
```

2. Update the following fields for a minimum unsecured running state (single node):

```
cluster.name: graylog
```

```
node.name: ${HOSTNAME}
```

```
path.data: /var/lib/opensearch path.logs:
/var/log/opensearch discovery.type:
single-node network.host: 0.0.0.0
action.auto_create_index: false
plugins.security.disabled: true
```

To create multi-node Opensearch clusters, see -

<https://opensearch.org/docs/latest/tuningyour-cluster/cluster/> 3. Enable JVM options:

```
sudo nano /etc/opensearch/jvm.options
```

4. Now, update the Xms & Xmx settings with half of the installed system memory

```
-Xms1g
```

```
-Xmx1g
```

5. Configure the kernel parameters at runtime: **sudo sysctl -w**

```
vm.max_map_count=262144 sudo echo
```

```
'vm.max_map_count=262144' >> /etc/sysctl.conf
```

6. Finally, enable the system service: **sudo systemctl**

```
daemon-reload sudo systemctl enable
```

```
opensearch.service sudo systemctl start
```

```
opensearch.service sudo systemctl status
```

```
opensearch.service
```

## Elasticsearch

Elasticsearch 7.10.2 is the only version that is compatible with Graylog 5.0; however, we recommend OpenSearch for new Graylog cluster installations.

1. The following commands will begin the installation of the open-source version of Elasticsearch. See the [Elasticsearch install page](#) for more detailed instructions.

```
wget -q https://artifacts.elastic.co/GPG-KEY-elasticsearch -O myKey sudo apt-key
add myKey echo "deb https://artifacts.elastic.co/packages/oss-7.x/apt stable main"
| sudo tee -a
```

```
/etc/apt/sources.list.d/elastic-7.x.list
```

```
sudo apt-get update && sudo apt-get install elasticsearch-oss
```

2. Modify the Elasticsearch configuration file (/etc/elasticsearch/elasticsearch.yml), set the cluster name to graylog, and uncomment action.auto\_create\_index: false to enable the action: **sudo tee -a /etc/elasticsearch/elasticsearch.yml > /dev/null <<EOT cluster.name: graylog action.auto\_create\_index: false**  
**EOT**

3. After you have modified the configuration, you can start Elasticsearch and verify it is running. **sudo systemctl daemon-reload sudo systemctl enable elasticsearch.service**

**sudo systemctl restart elasticsearch.service sudo systemctl --type=service --state=active | grep elasticsearch**

## Graylog

Install the Graylog Open repository configuration and Graylog itself with the following commands:

**wget https://packages.graylog2.org/repo/packages/graylog-5.0-repository\_latest.deb  
sudo dpkg -i graylog-5.0-repository\_latest.deb sudo apt-get update && sudo apt-get install graylog-server**

If you are installing Graylog Operations, then you will use the following commands:

**wget https://packages.graylog2.org/repo/packages/graylog-5.0-repository\_latest.deb sudo  
dpkg -i graylog-5.0-repository\_latest.deb  
sudo apt-get update && sudo apt-get install graylog-enterprise**

## Edit the Configuration File

Read the instructions within the configurations file and edit as needed, located at /etc/graylog/server/server.conf. Additionally add password\_secret and root\_password\_sha2 as these are mandatory and Graylog will not start without them.

1. To create your root\_password\_sha2, run the following command: **echo -n "Enter Password: " && head -1 </dev/stdin | tr -d '\n' | sha256sum | cut -d" " f1**

2. You will then need to use the following command to create your root\_password\_sha2:

**< /dev/urandom tr -dc A-Z-a-z-0-9 | head -c\${1:-96};echo;**

3. To be able to connect to Graylog, set http\_bind\_address to the public host name or a public IP address of the machine with which you can connect. More information about these settings can be found in [Configuring the web interface](#).

Hint: If you're operating a single-node setup and would like to use HTTPS for the Graylog web interface and the Graylog REST API, it's possible to use [NGINX or Apache as a reverse proxy](#).

4. The last step is to enable Graylog during the operating system's startup:

```
sudo systemctl daemon-reload sudo  
systemctl enable graylog-server.service sudo  
systemctl start graylog-server.service  
sudo systemctl --type=service --state=active | grep graylog
```

Now you can [ingest messages](#) into your new Graylog Cluster and extract the messages with [extractors](#) or use [pipelines](#) to work with the messages.

## PRACTICAL 10

### Demonstrate Conversion of Data into a Universal Format

#### Objectives:

Part 1: Normalize Timestamps in a Log File

Part 2: Normalize Timestamps in an Apache Log File

Part 3: Log File Preparation in Security Onion Virtual Machine

#### Theory:

This lab will prepare you to learn where log files are located and how to manipulate and view log files. Log entries are generated by network devices, operating systems, applications, and various types of programmable devices. A file containing a time-sequenced stream of log entries is called a log file.

By nature, log files record events that are relevant to the source. The syntax and format of data within log messages are often defined by the application developer

Therefore, the terminology used in the log entries often varies from source to source. For example, depending on the source, the terms login, logon, authentication event, and user connection, may all appear in log entries to describe a successful user authentication to a server.

It is often desirable to have a consistent and uniform terminology in logs generated by different sources. This is especially true when all log files are being collected by a centralized point

#### Steps:

Part 1: Normalize Timestamps in a Log File

Timestamps are used in log entries to specify when the recorded event took place. While it is best practice to record timestamps in UTC, the format of the timestamp varies from log source to log source. There are two common timestamp formats, known as Unix Epoch and Human Readable.

Unix Epoch timestamps record time by measuring the number of seconds that have passed since January 1, 1970.

Human Readable timestamps record time by representing separate values for year, month, day, hour, minute, and second.

The Human Readable Wed, 28 Jun 2017 13:27:19 GMT timestamp is the same as 1498656439 in Unix Epoch.



From a programmability standpoint, it is much easier to work with Epoch as it allows for easier addition and subtraction operations. From an analysis perspective; however, Human Readable timestamps are much easier to interpret.

### Converting Epoch to Human Readable Timestamps with AWK

AWK is a programming language designed to manipulate text files. It is very powerful and especially useful when handling text files where the lines contain multiple fields, separated by a delimiter character. Log files contain one entry per line and are formatted as delimiter-separated fields, making AWK a great tool for normalizing.

Consider the **applicationX\_in\_epoch.log** file below. The source of the log file is not relevant.

```
[analyst@secOps lab.support.files]$ cat applicationX_in_epoch.log
2|Z|1219071600|AF|0
3|N|1219158000|AF|89
4|N|1220799600|AS|12
1|Z|1220886000|AS|67
5|N|1220972400|EU|23
6|R|1221058800|OC|89

[analyst@secOps lab.support.files]$ S
```

The log file above was generated by what we will call application X. The relevant aspects of the file are:

The columns are separated, or delimited, by the `|` character. Therefore, the data has five columns.

The third column contains timestamps in Unix Epoch.

The file has an extra line at the end. This will be important later in the lab.

Assume that a log analyst needs to convert the timestamps to a human-readable format. Follow the steps below to use AWK to easily perform the manual conversion:

- Launch the **CyberOps Workstation VM** and then launch a terminal window.
- Use the `cd` command to change to the **/home/analyst/lab.support.files/** directory. A copy of the file shown above is stored there.

```
[analyst@secOps ~]$ cd /home/analyst/lab.support.files/
[analyst@secOps lab.support.files]$ ls -l
```

```

Terminal - analyst@secOps:~/lab.support.files
File Edit View Terminal Tabs Help
[analyst@secOps ~]$ cd lab.support.files/
[analyst@secOps lab.support.files]$ ls -l
total 580
-rw-r--r-- 1 analyst analyst 649 Mar 21 2018 apache_in_epoch.log
-rw-r--r-- 1 analyst analyst 126 Mar 21 2018 applicationX_in_epoch.log
drwxr-xr-x 4 analyst analyst 4096 Mar 21 2018 attack_scripts
-rw-r--r-- 1 analyst analyst 102 Mar 21 2018 confidential.txt
-rw-r--r-- 1 analyst analyst 2871 Mar 21 2018 cyops.mn
-rw-r--r-- 1 analyst analyst 75 Mar 21 2018 elk_services
-rw-r--r-- 1 analyst analyst 373 Mar 21 2018 h2_dropbear.banner
drwxr-xr-x 2 analyst analyst 4096 Apr 2 2018 instructor
-rw-r--r-- 1 analyst analyst 255 Mar 21 2018 letter_to_grandma.txt
-rw-r--r-- 1 analyst analyst 24464 Mar 21 2018 logstash-tutorial.log
drwxr-xr-x 2 analyst analyst 4096 Mar 21 2018 malware
-rwxr-xr-x 1 analyst analyst 172 Mar 21 2018 mininet_services
drwxr-xr-x 2 analyst analyst 4096 Mar 21 2018 openssl_lab
drwxr-xr-x 2 analyst analyst 4096 Mar 21 2018 pcaps
drwxr-xr-x 7 analyst analyst 4096 Mar 21 2018 pox
-rw-r--r-- 1 analyst analyst 473363 Mar 21 2018 sample.img
-rw-r--r-- 1 analyst analyst 65 Mar 21 2018 sample.img_SHA256.sig
drwxr-xr-x 3 analyst analyst 4096 Mar 21 2018 scripts
-rw-r--r-- 1 analyst analyst 25553 Mar 21 2018 SQL_Lab.pcap

```

c. Issue the following AWK command to convert and print the result on the terminal:

Note: Up arrow can be used to edit the typing errors in the previous command entry.

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS="|"} {$3=strftime("%c",$3)} {print}' applicationX_in_epoch.log
```

```

[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS="|"} {$3=strftime("%c",$3)} {print}' applicationX_in_epoch.log
2|Z|Mon 18 Aug 2008 11:00:00 AM EDT|AF|0
3|N|Tue 19 Aug 2008 11:00:00 AM EDT|AF|89
4|N|Sun 07 Sep 2008 11:00:00 AM EDT|AS|12
1|Z|Mon 08 Sep 2008 11:00:00 AM EDT|AS|67
5|N|Tue 09 Sep 2008 11:00:00 AM EDT|EU|23
6|R|Wed 10 Sep 2008 11:00:00 AM EDT|OC|89
||Wed 31 Dec 1969 07:00:00 PM EST
[analyst@secOps lab.support.files]$

```

The command above is an AWK script. It may seem complicated. The main structure of the AWK script above is as follows:

**awk** – This invokes the AWK interpreter.

**‘BEGIN** – This defines the beginning of the script.

**{}** – This defines actions to be taken in each line of the input text file. An AWK script can have several actions.

**FS = OFS = “|”** – This defines the field separator (i.e., delimiter) as the bar (|) symbol. Different text files may use different delimiting characters to separate fields. This operator allows the user to define what character is used as the field separator in the current text file.

**\$3** – This refers to the value in the third column of the current line. In the log, the third column contains the timestamp in epoch to be converted. **strftime** – This is an AWK internal function designed to work with time. The %c and \$3 in between parenthesis are the parameters passed to strftime.

**log** – This is the input text file to be loaded and used. Because you are already in the `lab.support.files` directory, you do not need to add path information, **/home/analyst/lab.support.files/applicationX\_in\_epoch.log**.

The first script action that defined in the first set of curly brackets is to define the field separator character as the “|”. Then, in the second set of curly brackets, it rewrites the third column of each line with the result of the execution of the **strftime()** function. **strftime()** is an internal AWK function created to handle time conversion. Notice that the script tells the function to use the contents of the third column of each line before the change (**\$3**) and to format the output (**%c**).

Were the Unix Epoch timestamps converted to Human Readable format? Were the other fields modified? Explain.

Yes, the script converted from Epoch to Human Readable. The script changed only the timestamp field, preserving the rest of the file.

Compare the contents of the file and the printed output. Why is there the line, **||Wed 31 Dec 1969 07:00:00 PM EST?**

The reason for the extra line is because the file has an empty line at the end, which led the script to mistakenly interpret it as 0 and convert that into a Human Readable timestamp.

By interpreting the empty line as 0, the script converted 0 Unix Epoch to Human Readable. 0 Unix Epoch translates to 0 seconds after midnight of Jan 1st, 1970. The script displays “Wed 31 Dec 1969 07:00:00 PM EST” because it automatically adjusts for the timezone. Because the CyberOps Workstation is configured for EST (UTC -5), the script displays midnight, Jan 1, 1970 minus 5 hours.

d. Use **nano** (or your favorite text editor) to remove the extra empty line at the end of the file and run the **AWK** script again by using the up-arrow to find it in the command history buffer.

```
[analyst@secOps lab.support.files]$ nano applicationX_in_epoch.log
```

Is the output correct now? Explain.

Yes. Because the empty line was removed, no extra data was created and added to the log file by the script.

e. While printing the result on the screen is useful for troubleshooting the script, analysts will likely need to save the output in a text file. Redirect the output of the script above to a file named `applicationX_in_human.log` to save it to a file:

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS="|"} {$3=strftime("%c", $3)} {print}' applicationX_in_epoch.log > applicationX_in_human.log  
[analyst@secOps lab.support.files]$
```

What was printed by the command above? Is this expected?

Nothing was printed on the screen. Yes, it is expected, as the command output was redirected to a text file named `applicationX_in_human.log`.

f. Use cat to view the applicationX\_in\_human.log. Notice that the extra line is now removed and the timestamps for the log entries have been converted to human readable format.

```
[analyst@secOps lab.support.files]$ cat applicationX_in_human.log
```

```
[analyst@secOps lab.support.files]$ cat applicationX_in_human.log
2|Z|Mon 18 Aug 2008 11:00:00 AM EDT|AF|0
3|N|Tue 19 Aug 2008 11:00:00 AM EDT|AF|89
4|N|Sun 07 Sep 2008 11:00:00 AM EDT|AS|12
1|Z|Mon 08 Sep 2008 11:00:00 AM EDT|AS|67
5|N|Tue 09 Sep 2008 11:00:00 AM EDT|EU|23
6|R|Wed 10 Sep 2008 11:00:00 AM EDT|OC|89
[analyst@secOps lab.support.files]$
```

## Part 2: Normalize Timestamps in an Apache Log File

Similar to what was done with the applicationX\_in\_epoch.log file, Apache web server log files can also be normalized. Follow the steps below to convert Unix Epoch to Human Readable timestamps. Consider the following Apache log file, **apache\_in\_epoch.log**:

```
[analyst@secOps lab.support.files]$ cat apache_in_epoch.log
```

```
[analyst@secOps lab.support.files]$ cat apache_in_epoch.log
198.51.100.213 - - [1219071600] "GET /twiki/bin/edit/Main/Double_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846
198.51.100.213 - - [1219158000] "GET /twiki/bin/rdiff/TWiki/NewUserTemplate?rev1=1.3&rev2=1.2 HTTP/1.1" 200 4523
198.51.100.213 - - [1220799600] "GET /mailman/listinfo/hsdivision HTTP/1.1" 200 6291
198.51.100.213 - - [1220886000] "GET /twiki/bin/view/TWiki/WikiSyntax HTTP/1.1" 200 7352
198.51.100.213 - - [1220972400] "GET /twiki/bin/view/Main/DCCAndPostFix HTTP/1.1" 200 5253
198.51.100.213 - - [1221058800] "GET /twiki/bin/oops/TWiki/AppendixFileSystem?template=oopsmore&m1=1.12&m2=1.12 HTTP/1.1" 200 11382
[analyst@secOps lab.support.files]$
```

The Apache Log file above contains six entries which record events related to the Apache web server. Each entry has seven fields. The fields are delimited by a space:

The first column contains the IPv4 address, **51.100.213**, of the web client placing the request.

The second and third columns are not used and a “-” character is used to represent no value.

The fourth column contains the timestamp in Unix Epoch time, for example **[1219071600]**.

The fifth column contains text with details about the event, including URLs and web request parameters. All six entries are HTTP GET messages. Because these messages include spaces, the entire field is enclosed with quotes.

The sixth column contains the HTTP status code, for example **401**.

The seventh column contains the size of the response to the client (in bytes), for example **12846**.

As in Part 1, a script will be created to convert the timestamp from Epoch to Human Readable.

a. First, answer the questions below. They are crucial for the construction of the script.

In the context of timestamp conversion, what character would work as a good delimiter character for the Apache log file above?

The space character

How many columns does the Apache log file above contain?

7

In the Apache log file above, what column contains the Unix Epoch Timestamp?

Column 4

b. In the **CyberOps Workstation VM** terminal, a copy of the Apache log file, `apache_in_epoch.log`, is stored in the `/home/analyst/lab.support.files`.

c. Use an **awk** script to convert the timestamp field to a human readable format. Notice that the command contains the same script used previously, but with a few adjustments for the delimiter, timestamp field, and file name.

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS=" "} {$4=strftime("%c",$4 )} {print}' apache_in_epoch.log
```

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS=" "} {$4=strftime("%c",$4 )} {print}' apache_in_epoch.log
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/edit/Main/Dou
ble_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/rdiff/Twiki/N
ewUserTemplate?rev1=1.3&rev2=1.2 HTTP/1.1" 200 4523
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /mailman/listinfo/hedivi
sion HTTP/1.1" 200 6291
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/view/TWiki/Wi
kiSyntax HTTP/1.1" 200 7352
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/view/Main/DCC
AndPostFix HTTP/1.1" 200 5253
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/oops/Twiki/Ap
pendixFileSystem?template=oopsmore&m1=1.12&m2=1.12 HTTP/1.1" 200 11382
[analyst@secOps lab.support.files]$
```

Was the script able to properly convert the timestamps? Describe the output.

No. All timestamps are now Wed 31 Dec 1969 07:00:00 PM EST. d.

Before moving forward, think about the output of the script.

Can you guess what caused the incorrect output? Is the script incorrect? What are the relevant differences between the **applicationX\_in\_epoch.log** and **apache\_in\_epoch.log**?

The problem is the square brackets in the course file. The script expects the timestamp to be in the Unix Epoch format which does not include the square brackets. Because the script does not know what number represents the "[" character, it assumes zero and returns the Unix beginning of time in UTC -5.



- e. To fix the problem, the square brackets must be removed from the timestamp field before the conversion takes place. Adjust the script by adding two actions before the conversion, as shown below:

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS=" "} {gsub(/[[]/, "", $4)}{print}}{$4=strftime("%c", $4)}{print}' apache_in_epoch.log
```

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS=" "} {gsub(/[[]/, "", $4)}{print}}{$4=strftime("%c", $4)}{print}'
apache_in_epoch.log
198.51.100.213 - - [1219071600] "GET /twiki/bin/edit/Main/Double_bounce_sender?topicparent=Main.ConfigurationVariable
s HTTP/1.1" 401 12846
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/edit/Main/Double_bounce_sender?topicparent=Main.Co
nfigurationVariables HTTP/1.1" 401 12846
198.51.100.213 - - [1219158000] "GET /twiki/bin/rdiff/TWiki/NewUserTemplate?rev1=1.3&rev2=1.2 HTTP/1.1" 200 4523
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/rdiff/TWiki/NewUserTemplate?rev1=1.3&rev2=1.2 HTTP
/1.1" 200 4523
198.51.100.213 - - [1220799600] "GET /mailman/listinfo/hsdivision HTTP/1.1" 200 6291
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /mailman/listinfo/hsdivision HTTP/1.1" 200 6291
198.51.100.213 - - [1220886000] "GET /twiki/bin/view/TWiki/WikiSyntax HTTP/1.1" 200 7352
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/view/TWiki/WikiSyntax HTTP/1.1" 200 7352
198.51.100.213 - - [1220972400] "GET /twiki/bin/view/Main/DCCAndPostFix HTTP/1.1" 200 5253
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/view/Main/DCCAndPostFix HTTP/1.1" 200 5253
198.51.100.213 - - [1221058800] "GET /twiki/bin/oops/TWiki/AppendixFileSystem?template=oopsmore&m1=1.12&m2=1.12 HTTP/
1.1" 200 11382
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/oops/TWiki/AppendixFileSystem?template=oopsmore&m1
=1.12&m2=1.12 HTTP/1.1" 200 11382
[analyst@secOps lab.support.files]$
```

Notice after specifying space as the delimiter with `{FS=OFS=" "}`, there is a regular expression action to match and replace the square brackets with an empty string, effectively removing the square brackets that appear in the timestamp field. The second action prints the updated line so the conversion action can be performed.

**gsub()** – This is an internal AWK function used to locate and substitute strings. In the script above, `gsub()` received three comma-separated parameters, described below.

`/[[]/` – This is a regular expression passed to **gsub()** as the first parameter. The regular expression should be read as **“find “[ OR “]”**. Below is the breakdown of the expression:

- The first and last `/` character marks the beginning and end of the search block. Anything between the first `/` and the second `/` are related to the search. The `“` character is used to escape the following `[`. Escaping is necessary because `[` can also be used by an operator in regular expressions. By escaping the `[` with a leading `“`, we tell the interpreter that the `]` is part of the content and not an operator. The `|` character is the OR operator. Notice that the `|` is not escaped and will therefore, be seen as an operator. Lastly, the regular expression escapes the closing square bracket with `”`, as done before.
- `“”` – This represents no characters, or an empty string. This parameter tells `gsub()` what to replace the `[` and `]` with, when found. By replacing the `[` and `]` with `“”`, `gsub()` effectively removes the `[` and `]` characters.
- `$4` – This tells `gsub()` to work only on the fourth column of the current line, the timestamp column.

Note: Regular expression interpretation is a SECOPS exam topic. Regular expressions are covered in more detail in another lab in this chapter. However, you may wish to search the Internet for tutorials.

- f. In a CyberOps Workstation VM terminal, execute the adjusted script, as follows:

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS=" "} {gsub(/[[|/],"/,"",$4)}{print}}{$4=strftime("%c",$4)}{print}' apache_in_epoch.log
```

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS=" "} {gsub(/[[|/],"/,"",$4)}{print}}{$4=strftime("%c",$4)}{print}' apache_in_epoch.log
198.51.100.213 - - [1219071600] "GET /twiki/bin/edit/Main/Double_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/edit/Main/Double_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846
198.51.100.213 - - [1219158000] "GET /twiki/bin/rdiff/TWiki/NewUserTemplate?rev1=1.3&rev2=1.2 HTTP/1.1" 200 4523
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/rdiff/TWiki/NewUserTemplate?rev1=1.3&rev2=1.2 HTTP/1.1" 200 4523
198.51.100.213 - - [1220799600] "GET /mailman/listinfo/hedivision HTTP/1.1" 200 6291
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /mailman/listinfo/hedivision HTTP/1.1" 200 6291
198.51.100.213 - - [1220886000] "GET /twiki/bin/view/TWiki/WikiSyntax HTTP/1.1" 200 7352
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/view/TWiki/WikiSyntax HTTP/1.1" 200 7352
198.51.100.213 - - [1220972400] "GET /twiki/bin/view/Main/DCCAndPostFix HTTP/1.1" 200 5253
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/view/Main/DCCAndPostFix HTTP/1.1" 200 5253
198.51.100.213 - - [1221058800] "GET /twiki/bin/oops/TWiki/AppendixFileSystem?template=oopsmore&m1=1.12&m2=1.12 HTTP/1.1" 200 11382
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/oops/TWiki/AppendixFileSystem?template=oopsmore&m1=1.12&m2=1.12 HTTP/1.1" 200 11382
[analyst@secOps lab.support.files]$
```

Was the script able to properly convert the timestamps this time? Describe the output.

Yes. The output now displays two lines for each log entry. The first line displays the timestamp in Unix Epoch format and the second line is the same log entry with the timestamp displayed using Human Readable format.

g. Shut down CyberOps Workstation VM if desired.

## Part 3: Log File Preparation in Security Onion Virtual Machine

Because log file normalization is important, log analysis tools often include log normalization features. Tools that do not include such features often rely on plugins for log normalization and preparation. The goal of these plugins is to allow log analysis tools to normalize and prepare the received log files for tool consumption.

The Security Onion appliance relies on a number of tools to provide log analysis services.

**ELK, Zeek, Snort and SGUIL** are arguably the most used tools.

**ELK** (Elasticsearch, Logstash, and Kibana) is a solution to achieve the following:

Normalize, store, and index logs at unlimited volumes and rates.

Provide a simple and clean search interface and API.

Provide an infrastructure for alerting, reporting and sharing logs.

Plugin system for taking actions with logs.

Exist as a completely free and open-source project.

**Zeek** (formerly called Bro) is a framework designed to analyze network traffic passively and generate event logs based on it. Upon network traffic analysis, Zeek creates logs describing events such as the following:

TCP/UDP/ICMP network connections

DNS activity



FTP activity

HTTPS requests and replies

SSL/TLS handshakes

### Snort and SGUIL

Snort is an IDS that relies on pre-defined rules to flag potentially harmful traffic. Snort looks into all portions of network packets (headers and payload), looking for patterns defined in its rules. When found, Snort takes the action defined in the same rule.

SGUIL provides a graphical interface for Snort logs and alerts, allowing a security analyst to pivot from SGUIL into other tools for more information. For example, if a potentially malicious packet is sent to the organization web server and Snort raised an alert about it, SGUIL will list that alert. The analyst can then right-click that alert to search the ELSA or Bro databases for a better understanding of the event.

Note: The directory listing maybe different than the sample output shown below.

Step 1: Start Security **Onion VM**.

Launch the **Security Onion VM** from VirtualBox's Dashboard (username: **analyst** / password: **cyberops**).

Step 2: Zeek Logs in Security Onion

- a. Open a terminal window in the Security Onion VM. Right-click the Desktop. In the pop-up menu, select Open Terminal.
- b. Zeek logs are stored at **/nsm/bro/logs/**. As usual with Linux systems, log files are rotated based on the date, renamed and stored on the disk. The current log files can be found under the current directory. From the terminal window, change directory using the following command.

```
analyst@SecOnion:~$ cd /nsm/bro/logs/current
analyst@SecOnion:/nsm/logs/current$
```

- c. Use the **ls -l** command to see the log files generated by Zeek:

Note: Depends on the state of the virtual machine, there may not be any log files yet.

Step 3: Snort Logs

- a. Snort logs can be found at **/nsm/sensor\_data/**. Change directory as follows.

```
analyst@SecOnion:/nsm/bro/logs/current$ cd /nsm/sensor_data
analyst@SecOnion:/nsm/sensor_data$
```

- b. Use the **ls -l** command to see all the log files generated by Snort.

```
analyst@SecOnion:/nsm/sensor_data$ ls -l
total 12
```

- c. Notice that Security Onion separates files based on the interface. Because the Security Onion VM image has two interfaces configured as sensors and a special folder for imported data, three directories are kept. Use the `ls -l seconion-eth0` command to see the files generated by the eth0 interface.

```
analyst@SecOnion:/nsm/sensor_data$ ls -l seconion-eth0
total 28 drwxrwxr-x 2 sguil sguil 4096 Jun 19 18:09
argus drwxr-xr-x 2 sguil sguil 4096 Jun 19 18:24
snort-1
```

#### Step 4: Various Logs

While the `/nsm/` directory stores some log files, more specific log files can be found under `/var/log/nsm/`. Change directory and use the `ls` command to see all the log files in the directory.

```
analyst@SecOnion:/nsm/sensor_data$ cd /var/log/nsm/
analyst@SecOnion:/var/log/nsm$ ls
sid_changes.log  netsniff-sync.log  so-elastic-configure-kibana-
dashboards.log  ossec_agent.log    so-elasticsearch-pipelines.log
pulledpork.log   so-sensor-backup-config.log  seconion-eth0
so-server-backup-config.log  seconion-import  so-setup.log
securityonion  so-zeek-cron.log  sensor-clean.log
squert-ip2c-5min.log  sensor-clean.log.1.gz  squert-ip2c.log  sensor-
clean.log.2.gz      squert_update.log  sensor-newday-argus.log
watchdog.log  sensor-newday-http-agent.log  watchdog.log.1.gz  sensor-
newday-pcap.log    watchdog.log.2.gz  sguil-db-purge.log
```

Notice that the directory shown above also contains logs used by secondary tools such as OSSEC and Squert.

b. ELK logs can be found in the /var/log Change directory and use the ls command to list the files and directories.

```
analyst@SecOnion:/var/log/nsm$ cd ..
analyst@SecOnion:/var/log$ ls
alternatives.log          debug
kern.log.1               samba alternatives.log.1  debug.1          kern.log.2.gz
sguild apache2           debug.2.gz               kibana           so-boot.log
apt                      dmesg                   lastlog          syslog auth.log
domain_stats             lightdm                 syslog.1 auth.log.1      dpkg.log
logstash                 syslog.2.gz auth.log.2.gz      dpkg.log.1      lpr.log
syslog.3.gz boot          elastalert              mail.err          syslog.4.gz
boot.log                 elasticsearch           mail.info         unattended-upgrades
bootstrap.log            error                   mail.log          user.log btmp
error.1                  mail.warn              user.log.1 btmp.1            error.2.gz
messages                 user.log.2.gz cron.log           faillog           messages.1
wtmp cron.log.1          freq_server             messages.2.gz wtmp.1 cron.log.2.gz
freq_server_dns          mysql                  Xorg.0.log curator          fsck
nsm                      Xorg.0.log.old daemon.log          gpu-manager.log  ntpstats
daemon.log.1             installer              redis daemon.log.2.gz kern.log
salt
```

c. Take some time to Google these secondary tools and answer the questions below:

For each one of the tools listed above, describe the function, importance, and placement in the security analyst workflow.

Pulledpork is a Snort rule manage system. It facilitates Snort rules updating. Outdated Snort rules makes the entire system useless.

OSSEC is a system used to normalize and concentrate local system logs. When deployed throughout the organization, OSSEC allows an analyst to have a clear picture of what is happening in the systems.

Squert is a visual tool that attempts to provide additional context to events through the use of metadata, time series representations, and weighted and logically grouped result sets.

Elasticsearch is a distributed search and analytics engine. The data is stored centrally to provide fast searches and allow for fine tuning.

Logstash gathers the data from different sources and feeds the data into ElasticSearch.

Kibana is the data visualization for the ELK stack to provide quick insight into the data.

#### Reflection

Log normalization is important and depends on the deployed environment.

Popular tools include their own normalization features, but log normalization can also be done manually.

When manually normalizing and preparing log files, double-check scripts to ensure the desired result is achieved. A poorly written normalization script may modify the data, directly impacting the analyst's work.