JHU SU20 IDS Module 4 Lab

Audrey Long

06/20/2020

1. What is Snort?
Start by reading these two items:
https://www.snort.org/faq/what-is-snort
https://en.wikipedia.org/wiki/Snort_(software).

Now that we're getting away from the host-based IDS's and into the network-based IDS's,

1.) **Describe how Snort can be a useful tool for detecting network attacks and how it can be customized to detect new attacks.**

Snort is an open source network intrusion prevention system, capable of performing real-time traffic analysis and packet logging on IP networks. It can perform protocol analysis, content searching/matching, and can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, and much more.

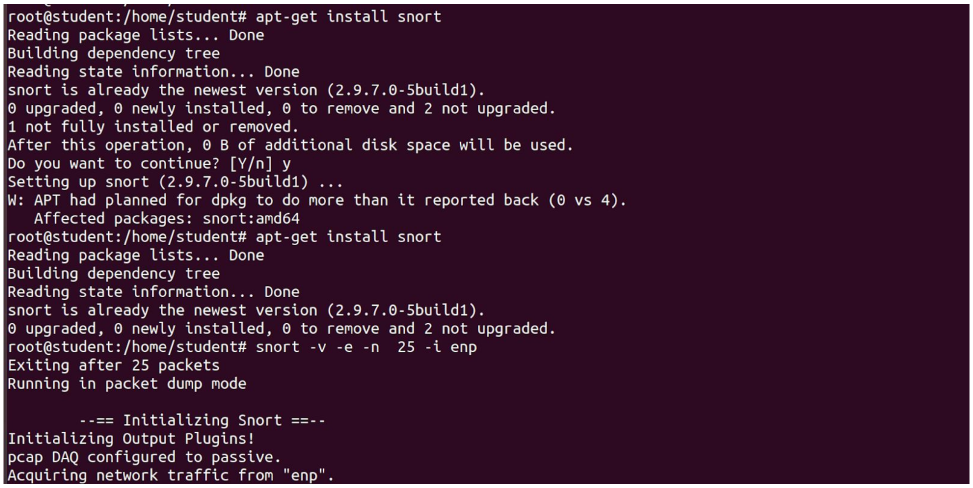
One of the important attacks that Snort detects is port scanning. Attackers commonly attempt to connect to other hosts and scan their ports as starters to other attacks. Using this technique, the attacker tries to identify the existence of hosts on a network or whether a particular service is in use. Such services include email, telnet, file transfer, HTTP, and DNS. Since a port is the Interface for each service within a computer, the information goes in and out of a computer through this port.

Snort Intrusion Detection System (Snort-IDS) is a security tool of network security. It has been widely used for protecting the network of the organizations. The Snort-IDS utilize the rules to match data packet traffic. If some packet matches the rules, Snort-IDS will generate the alert messages. This feature of Snort can be customizable to introduce new rules to detect future network attacks

2. Snort Options

1.) **If it is not already installed on your Ubuntu VM, run the following command to install Snort: sudo apt-get install snort.**

The picture below shows snort was configured with enp0s3 instead of the eth0 port.



```
root@student:/home/student# apt-get install snort
Reading package lists... Done
Building dependency tree
Reading state information... Done
snort is already the newest version (2.9.7.0-5build1).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
1 not fully installed or removed.
After this operation, 0 B of additional disk space will be used.
Do you want to continue? [Y/n] y
Setting up snort (2.9.7.0-5build1) ...
W: APT had planned for dpkg to do more than it reported back (0 vs 4).
   Affected packages: snort:amd64
root@student:/home/student# apt-get install snort
Reading package lists... Done
Building dependency tree
Reading state information... Done
snort is already the newest version (2.9.7.0-5build1).
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
root@student:/home/student# snort -v -e -n  25 -i enp
Exiting after 25 packets
Running in packet dump mode

        --== Initializing Snort ==--
Initializing Output Plugins!
pcap DAQ configured to passive.
Acquiring network traffic from "enp".
```

Figure 1: Snort installation

2.) **Consider the following Snort command: snort -v -e -n 25 -i eth0 -A fast. Describe what each of the flags in this command do. Instead of reading input from an interface, what command would you use to run Snort against an existing PCAP file?**

The following command parameters mean the following in the Snort man page:

-v: Be verbose. Prints packets out to the console. There is one big problem with verbose mode: it's slow. If you are doing IDS work with Snort, **don't** use the '-v' switch, you **WILL** drop packets.

-e: Display/log the link layer packet headers.

-i: interface Sniff packets on *interface.*

-A: alert-mode

Alert using the specified *alert-mode.* Valid alert modes include **fast, full, none,** and **unsock. Fast** writes alerts to the default "alert" file in a single-line, syslog style alert message. **Full** writes the alert to the "alert" file with the full decoded header as well as the alert message. **None** turns off alerting. **Unsock** is an experimental mode that sends the alert information out over a UNIX socket to another process that attaches to that Socket.

According to the man pages I would probably use the following commands to analyze an existing PCAP file the commands below show how to read a single PCAP file :

--pcap-single=*tcpdump-file*

Same as -r. Added for completeness.

AND -r is:

-r tcpdump-file

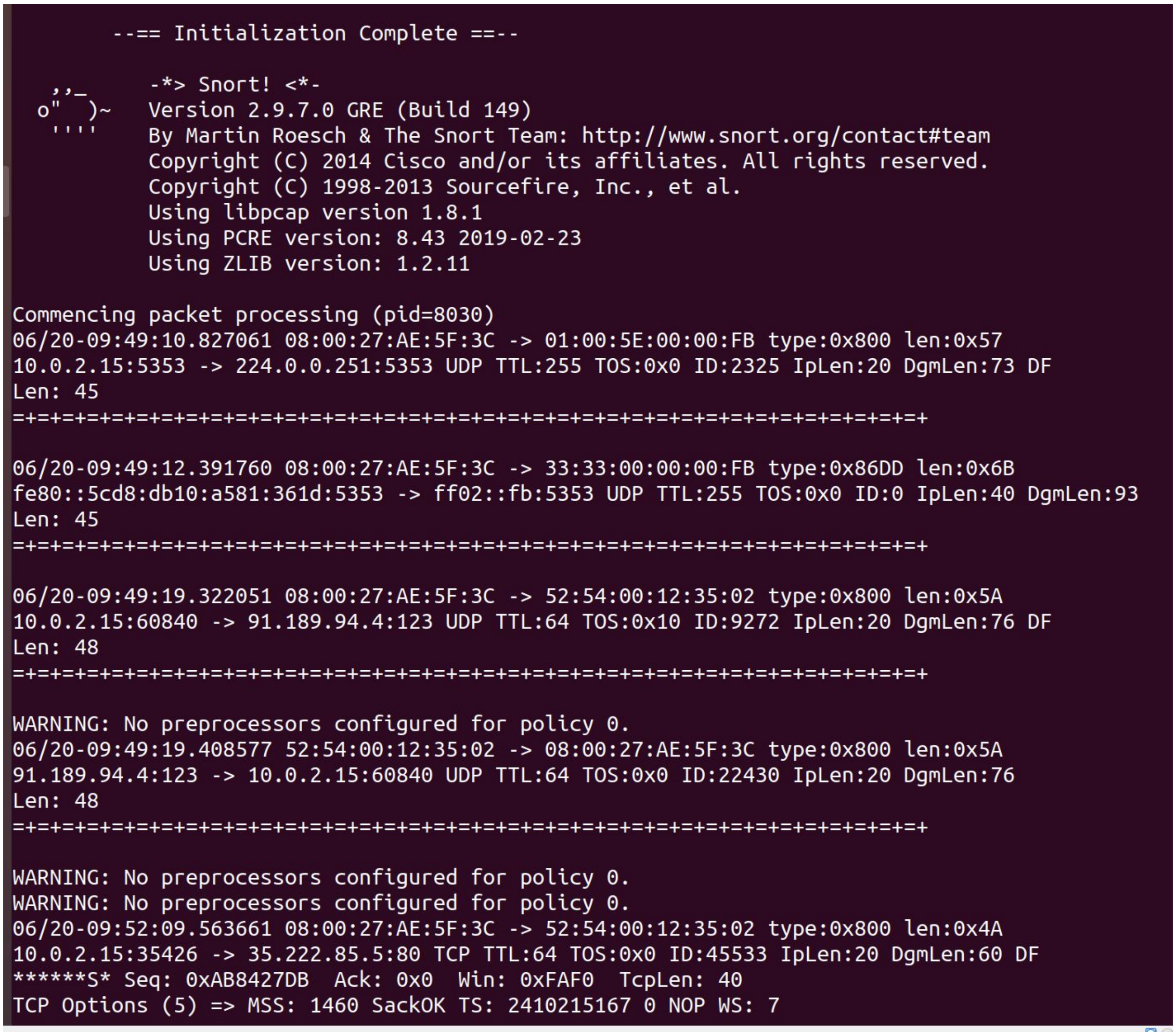
Read the tcpdump-formatted file *tcpdump-file.* This will cause Snort to read and process the file fed to it. This is useful if, for instance, you've got a bunch of SHADOW files that you want to process for content, or even if you've got a bunch of reassembled packet fragments which have been written into a tcp-dump formatted file.

--pcap-single=*tcpdump-file*

Same as -r. Added for completeness.

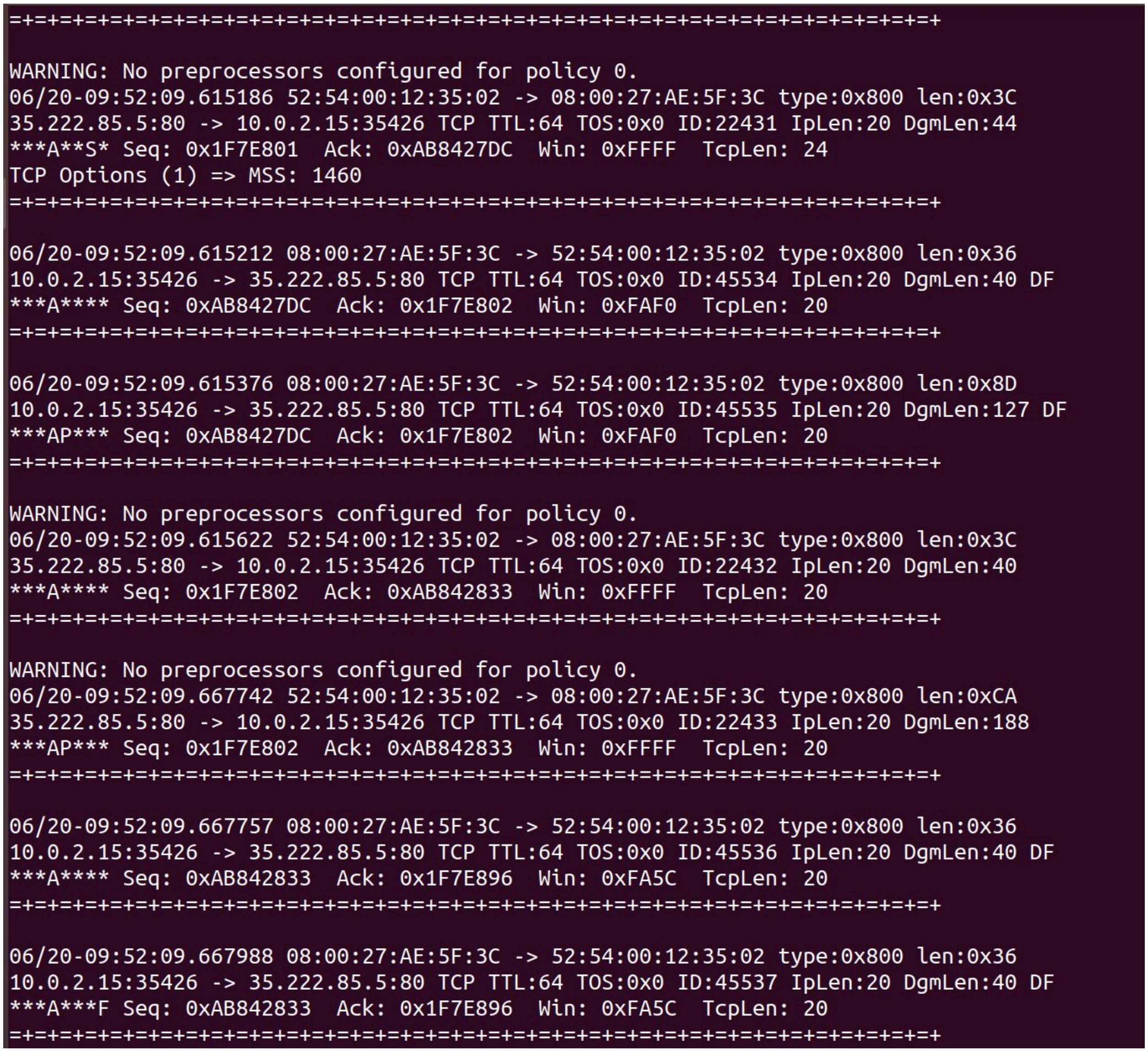
Read a single pcap

```
$ snort -r foo.pcap
$ snort --pcap-single=foo.pcap
```



```
        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.7.0 GRE (Build 149)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014 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: 8.43 2019-02-23
           Using ZLIB version: 1.2.11

Commencing packet processing (pid=8030)
06/20-09:49:10.827061 08:00:27:AE:5F:3C -> 01:00:5E:00:00:FB type:0x800 len:0x57
10.0.2.15:5353 -> 224.0.0.251:5353 UDP TTL:255 TOS:0x0 ID:2325 IpLen:20 DgmLen:73 DF
Len: 45
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/20-09:49:12.391760 08:00:27:AE:5F:3C -> 33:33:00:00:00:FB type:0x86DD len:0x6B
fe80::5cd8:db10:a581:361d:5353 -> ff02::fb:5353 UDP TTL:255 TOS:0x0 ID:0 IpLen:40 DgmLen:93
Len: 45
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/20-09:49:19.322051 08:00:27:AE:5F:3C -> 52:54:00:12:35:02 type:0x800 len:0x5A
10.0.2.15:60840 -> 91.189.94.4:123 UDP TTL:64 TOS:0x10 ID:9272 IpLen:20 DgmLen:76 DF
Len: 48
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
06/20-09:49:19.408577 52:54:00:12:35:02 -> 08:00:27:AE:5F:3C type:0x800 len:0x5A
91.189.94.4:123 -> 10.0.2.15:60840 UDP TTL:64 TOS:0x0 ID:22430 IpLen:20 DgmLen:76
Len: 48
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
WARNING: No preprocessors configured for policy 0.
06/20-09:52:09.563661 08:00:27:AE:5F:3C -> 52:54:00:12:35:02 type:0x800 len:0x4A
10.0.2.15:35426 -> 35.222.85.5:80 TCP TTL:64 TOS:0x0 ID:45533 IpLen:20 DgmLen:60 DF
******S* Seq: 0xAB8427DB  Ack: 0x0  Win: 0xFAF0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 2410215167 0 NOP WS: 7
```

Figure 2: Running the Snort command *“snort -v -e -n 25 -i enp0s3 -A fast”*



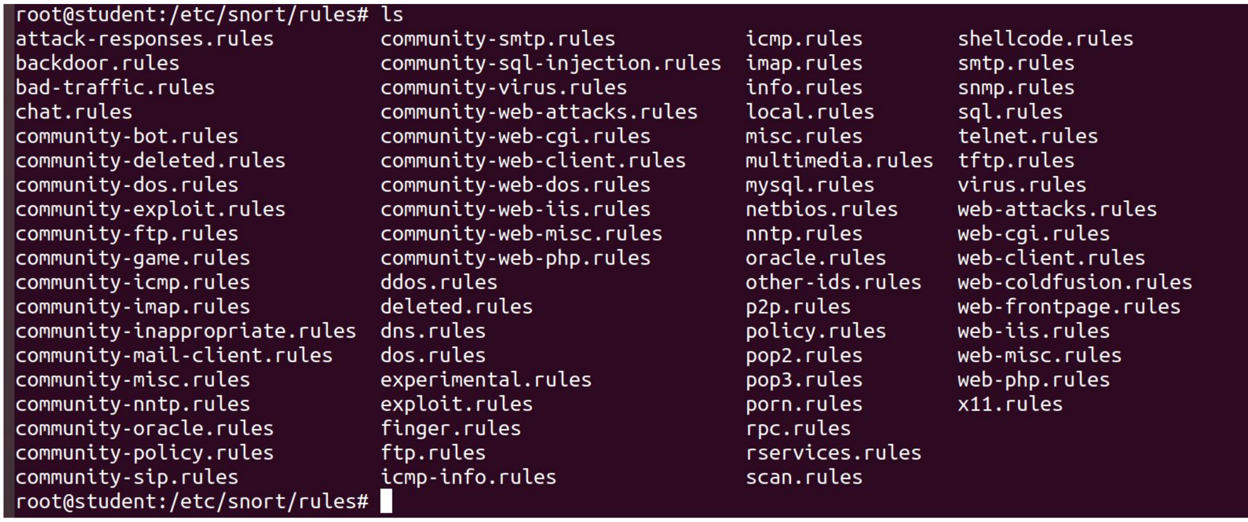
```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
06/20-09:52:09.615186 52:54:00:12:35:02 -> 08:00:27:AE:5F:3C type:0x800 len:0x3C
35.222.85.5:80 -> 10.0.2.15:35426 TCP TTL:64 TOS:0x0 ID:22431 IpLen:20 DgmLen:44
***A**S* Seq: 0x1F7E801  Ack: 0xAB8427DC  Win: 0xFFFF  TcpLen: 24
TCP Options (1) => MSS: 1460
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/20-09:52:09.615212 08:00:27:AE:5F:3C -> 52:54:00:12:35:02 type:0x800 len:0x36
10.0.2.15:35426 -> 35.222.85.5:80 TCP TTL:64 TOS:0x0 ID:45534 IpLen:20 DgmLen:40 DF
***A**** Seq: 0xAB8427DC  Ack: 0x1F7E802  Win: 0xFAF0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/20-09:52:09.615376 08:00:27:AE:5F:3C -> 52:54:00:12:35:02 type:0x800 len:0x8D
10.0.2.15:35426 -> 35.222.85.5:80 TCP TTL:64 TOS:0x0 ID:45535 IpLen:20 DgmLen:127 DF
***AP*** Seq: 0xAB8427DC  Ack: 0x1F7E802  Win: 0xFAF0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
06/20-09:52:09.615622 52:54:00:12:35:02 -> 08:00:27:AE:5F:3C type:0x800 len:0x3C
35.222.85.5:80 -> 10.0.2.15:35426 TCP TTL:64 TOS:0x0 ID:22432 IpLen:20 DgmLen:40
***A**** Seq: 0x1F7E802  Ack: 0xAB842833  Win: 0xFFFF  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

WARNING: No preprocessors configured for policy 0.
06/20-09:52:09.667742 52:54:00:12:35:02 -> 08:00:27:AE:5F:3C type:0x800 len:0xCA
35.222.85.5:80 -> 10.0.2.15:35426 TCP TTL:64 TOS:0x0 ID:22433 IpLen:20 DgmLen:188
***AP*** Seq: 0x1F7E802  Ack: 0xAB842833  Win: 0xFFFF  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/20-09:52:09.667757 08:00:27:AE:5F:3C -> 52:54:00:12:35:02 type:0x800 len:0x36
10.0.2.15:35426 -> 35.222.85.5:80 TCP TTL:64 TOS:0x0 ID:45536 IpLen:20 DgmLen:40 DF
***A**** Seq: 0xAB842833  Ack: 0x1F7E896  Win: 0xFA5C  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/20-09:52:09.667988 08:00:27:AE:5F:3C -> 52:54:00:12:35:02 type:0x800 len:0x36
10.0.2.15:35426 -> 35.222.85.5:80 TCP TTL:64 TOS:0x0 ID:45537 IpLen:20 DgmLen:40 DF
***A***F Seq: 0xAB842833  Ack: 0x1F7E896  Win: 0xFA5C  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Figure 3: Running the Snort command *“snort -v -e -n 25 -i enp0s3 -A fast”*

3. Snort Filters

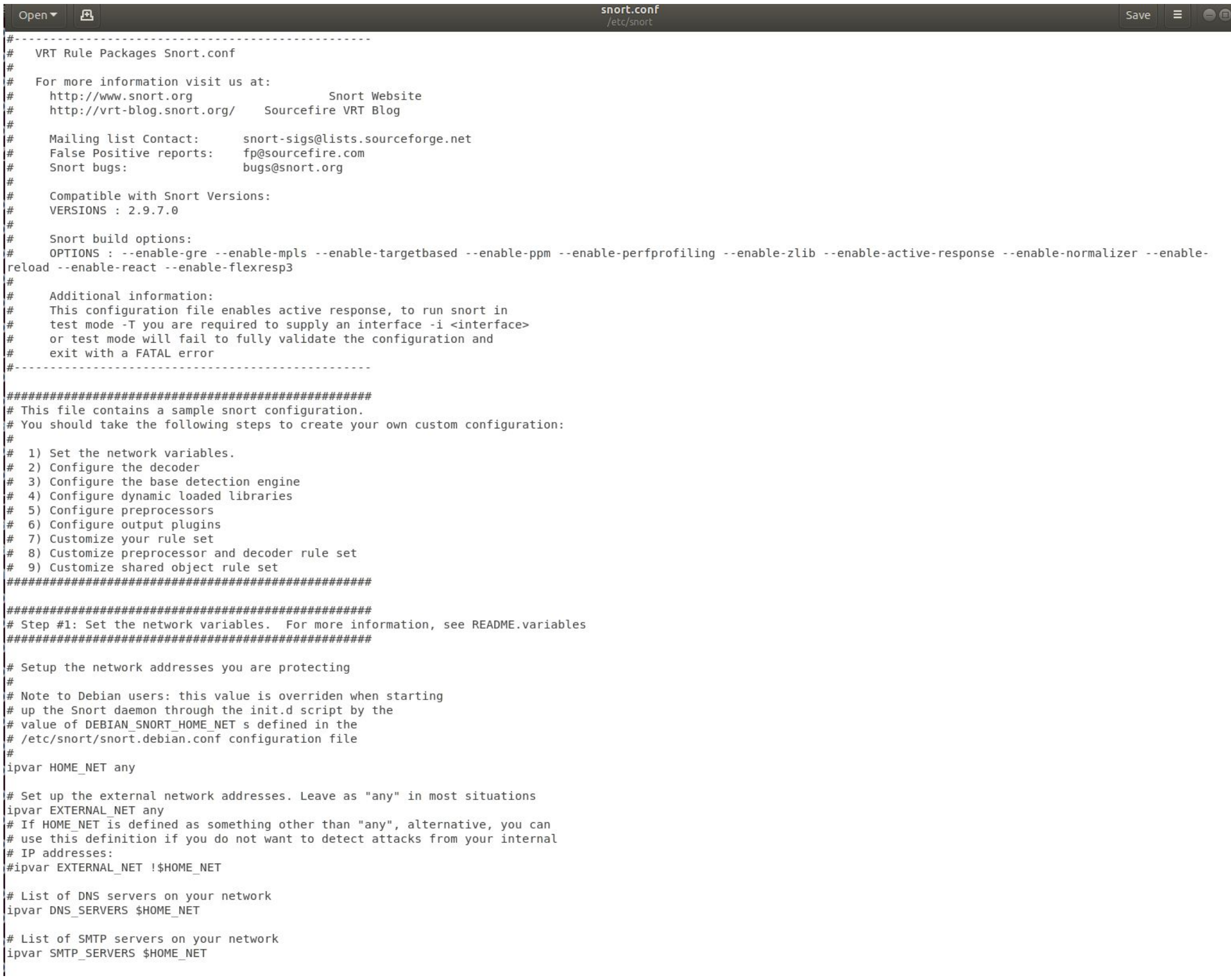
As we mentioned in the first section, Snort rules are highly configurable. Snort can use a common filtering protocol known as the Berkeley Packet Filter (BPF). In this exercise you will decode a Snort rule as well as write your own!

1.) First, which configuration file would you add custom Snort rules to or view to see a list of existing rules?



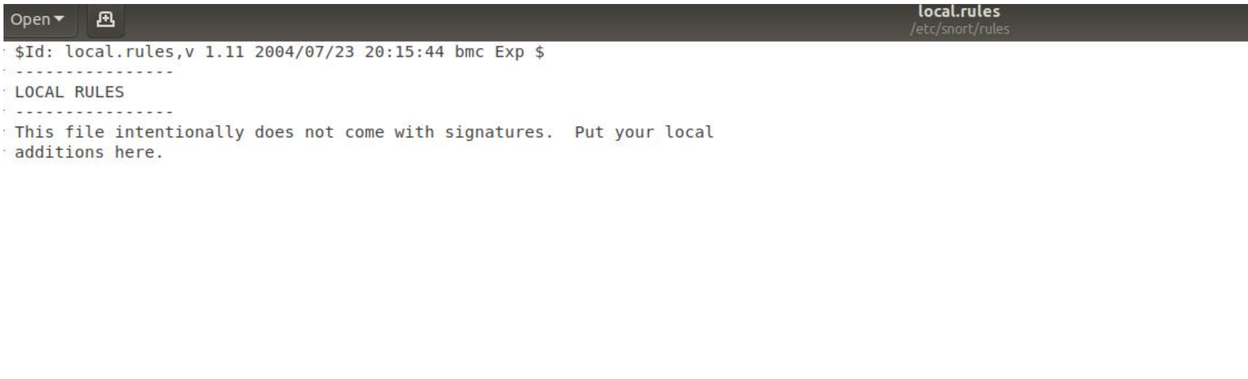
```
root@student:/etc/snort/rules# ls
attack-responses.rules          community-smtp.rules            icmp.rules         shellcode.rules
backdoor.rules                  community-sql-injection.rules   imap.rules         smtp.rules
bad-traffic.rules               community-virus.rules           info.rules         snmp.rules
chat.rules                      community-web-attacks.rules     local.rules        sql.rules
community-bot.rules             community-web-cgi.rules         misc.rules         telnet.rules
community-deleted.rules         community-web-client.rules      multimedia.rules   tftp.rules
community-dos.rules             community-web-dos.rules         mysql.rules        virus.rules
community-exploit.rules         community-web-iis.rules         netbios.rules      web-attacks.rules
community-ftp.rules             community-web-misc.rules        nntp.rules         web-cgi.rules
community-game.rules            community-web-php.rules         oracle.rules       web-client.rules
community-icmp.rules            ddos.rules                      other-ids.rules    web-coldfusion.rules
community-imap.rules            deleted.rules                   p2p.rules          web-frontpage.rules
community-inappropriate.rules   dns.rules                       policy.rules       web-iis.rules
community-mail-client.rules     dos.rules                       pop2.rules         web-misc.rules
community-misc.rules            experimental.rules              pop3.rules         web-php.rules
community-nntp.rules            exploit.rules                   porn.rules         x11.rules
community-oracle.rules          finger.rules                    rpc.rules
community-policy.rules          ftp.rules                       rservices.rules
community-sip.rules             icmp-info.rules                 scan.rules
root@student:/etc/snort/rules#
```

Figure 4: List of existing Snort rules



```
#--------------------------------------------------
#   VRT Rule Packages Snort.conf
#
#   For more information visit us at:
#     http://www.snort.org                   Snort Website
#     http://vrt-blog.snort.org/    Sourcefire VRT Blog
#
#     Mailing list Contact:      snort-sigs@lists.sourceforge.net
#     False Positive reports:    fp@sourcefire.com
#     Snort bugs:                bugs@snort.org
#
#     Compatible with Snort Versions:
#     VERSIONS : 2.9.7.0
#
#     Snort build options:
#     OPTIONS : --enable-gre --enable-mpls --enable-targetbased --enable-ppm --enable-perfprofiling --enable-zlib --enable-active-response --enable-normalizer --enable-
reload --enable-react --enable-flexresp3
#
#     Additional information:
#     This configuration file enables active response, to run snort in
#     test mode -T you are required to supply an interface -i <interface>
#     or test mode will fail to fully validate the configuration and
#     exit with a FATAL error
#--------------------------------------------------

###################################################
# This file contains a sample snort configuration.
# You should take the following steps to create your own custom configuration:
#
#  1) Set the network variables.
#  2) Configure the decoder
#  3) Configure the base detection engine
#  4) Configure dynamic loaded libraries
#  5) Configure preprocessors
#  6) Configure output plugins
#  7) Customize your rule set
#  8) Customize preprocessor and decoder rule set
#  9) Customize shared object rule set
###################################################

###################################################
# Step #1: Set the network variables.  For more information, see README.variables
###################################################

# Setup the network addresses you are protecting
#
# Note to Debian users: this value is overriden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET s defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET any

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
# If HOME_NET is defined as something other than "any", alternative, you can
# use this definition if you do not want to detect attacks from your internal
# IP addresses:
#ipvar EXTERNAL_NET !$HOME_NET

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET
```

Figure 5: snort.conf file to customize rule set



```
$Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
----------------
LOCAL RULES
----------------
This file intentionally does not come with signatures.  Put your local
additions here.
```

Figure 6: local.rules file to put in custom rules

2.) **Next, describe what the following rule does: alert ip any any - > 192.168.40.4080 (msg: ”Web traffic detected.”;).**

Alert : shows that this rule will generate an alert message when the criteria are met for a captured packet. The criteria are defined by the words that follow.

Ip: This part shows that this rule will be applied on all *IP* packets.

Any: is used for source *IP* address and shows that the rule will be applied to all packets.

Any: is used for the port number. Since port numbers are irrelevant at the *IP* layer, the rule will be applied to all packets.

-> : sign shows the direction of the packet.

192.168.40: The destination *IP* address and shows that the rule will be applied to all packets irrespective of destination *IP* address.

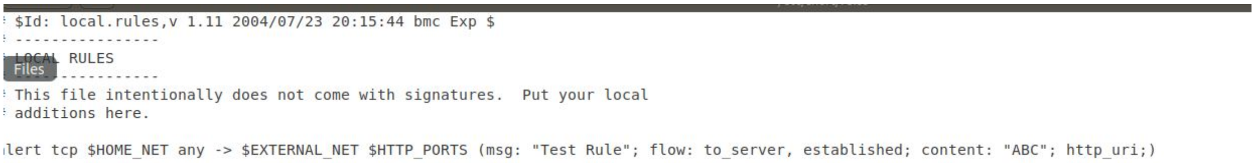
4080: destination port

(msg: ”Web traffic detected.”;) : The last part is the rule options and contains a message that will be logged along with the alert.

The rule will generate an alert message for *every* captured *IP* packet captured from any source address and shows that the rule will be applied to all packets irrespective of destination *IP* address, and generates the message provided.

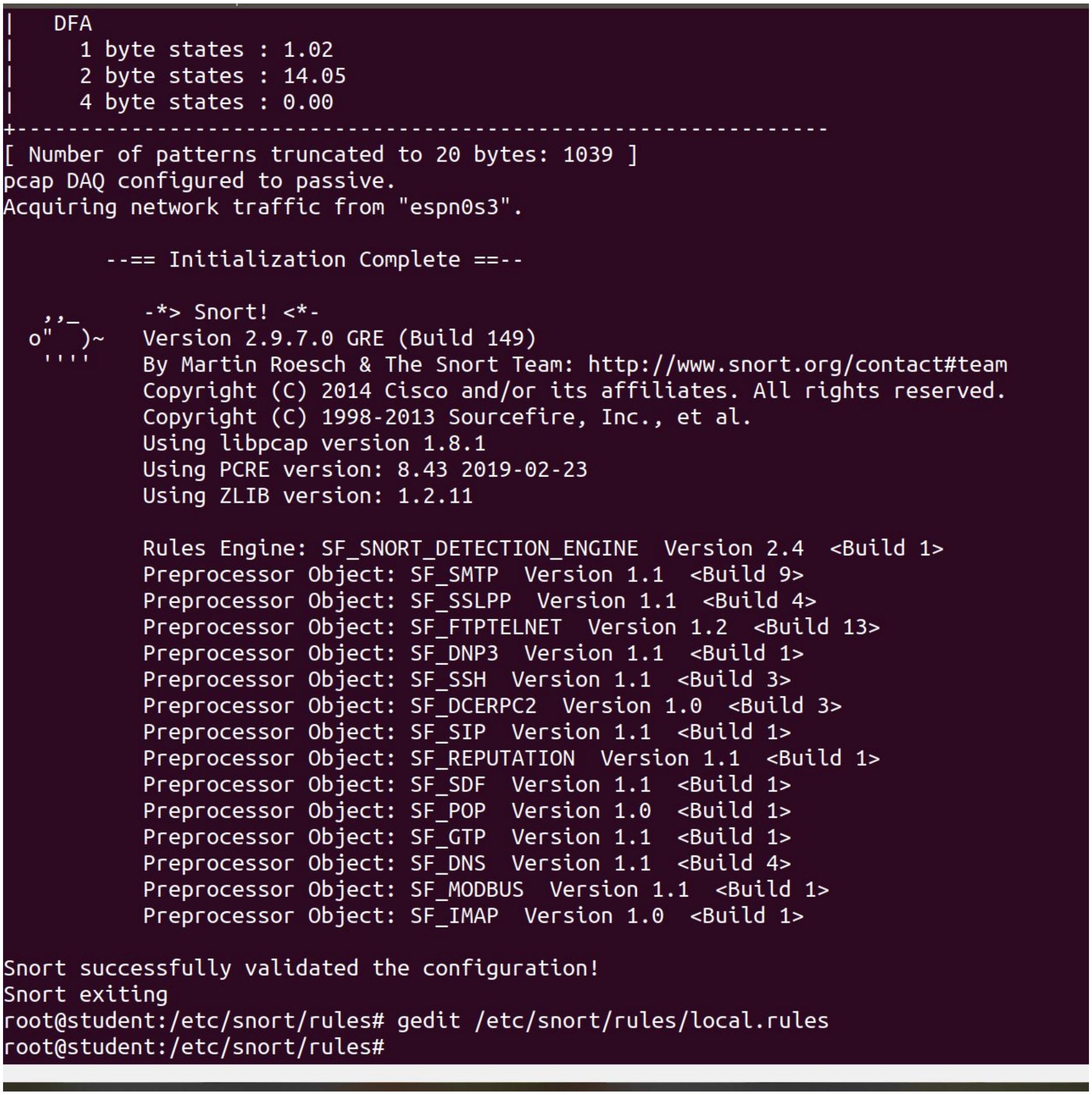
3.) **Next, write a rule that generates an alert whenever an internal network, from any originating port, connects to an external server on any of the standard (both encrypted an unencrypted) HTTP(S) ports. The alert can have a custom error message, but please be sure to include the HTTP URI**

of the remote host in the alert as well. Note: Uniform Resource Indicator (URI), URL is a form of URI which expresses an address which maps onto an access algorithm using network protocols. e.g. content; “/”; http(underscore)uri; after “established” to include HTTP URI .



```
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# ----------------
# LOCAL RULES
# ----------------
# This file intentionally does not come with signatures.  Put your local
# additions here.

lert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg: "Test Rule"; flow: to_server, established; content: "ABC"; http_uri;)
```

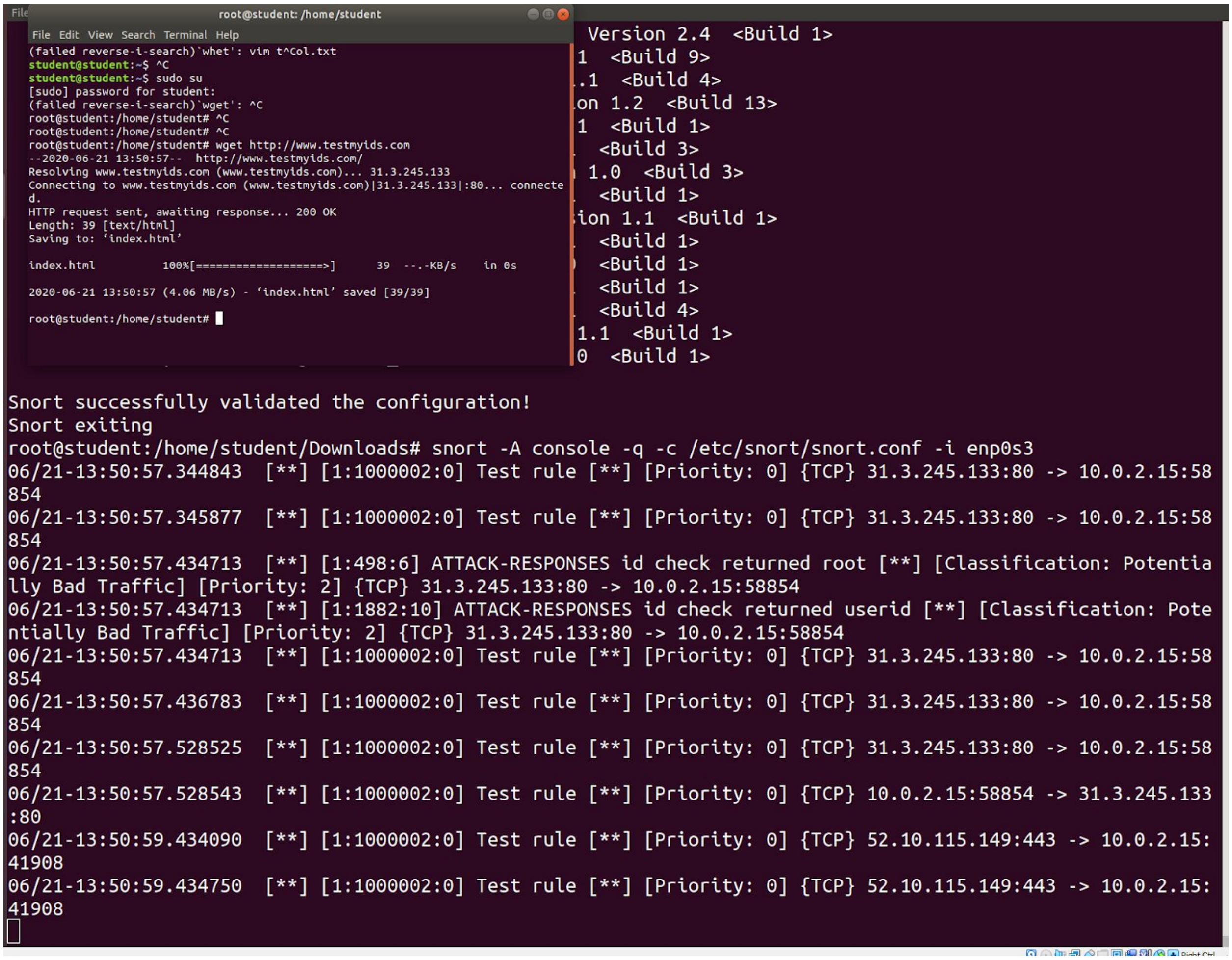
Figure 7: example rule added to the local rules file.



```
|   DFA
|     1 byte states : 1.02
|     2 byte states : 14.05
|     4 byte states : 0.00
+-------------------------------------------------------------
[ Number of patterns truncated to 20 bytes: 1039 ]
pcap DAQ configured to passive.
Acquiring network traffic from "espn0s3".

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.7.0 GRE (Build 149)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014 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: 8.43 2019-02-23
           Using ZLIB version: 1.2.11

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 2.4  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>

Snort successfully validated the configuration!
Snort exiting
root@student:/etc/snort/rules# gedit /etc/snort/rules/local.rules
root@student:/etc/snort/rules#
```

Figure 8: confirmation of valid configuration file rule added

I validated the rule by running “snort -T -i espn0s3 -c /etc/snort/snort.conf” to ensure there were no errors in the snort configuration file.


root@student: /home/student
File Edit View Search Terminal Help

```
(failed reverse-i-search)`whet': vim t^Col.txt
student@student:~$ ^C
student@student:~$ sudo su
[sudo] password for student:
(failed reverse-i-search)`wget': ^C
root@student:/home/student# ^C
root@student:/home/student# ^C
root@student:/home/student# wget http://www.testmyids.com
--2020-06-21 13:50:57--  http://www.testmyids.com/
Resolving www.testmyids.com (www.testmyids.com)... 31.3.245.133
Connecting to www.testmyids.com (www.testmyids.com)|31.3.245.133|:80... connecte
d.
HTTP request sent, awaiting response... 200 OK
Length: 39 [text/html]
Saving to: ‘index.html’

index.html          100%[===================>]      39  --.-KB/s    in 0s

2020-06-21 13:50:57 (4.06 MB/s) - ‘index.html’ saved [39/39]

root@student:/home/student# 
```

```
 Version 2.4  <Build 1>
1  <Build 9>
.1  <Build 4>
on 1.2  <Build 13>
1  <Build 1>
  <Build 3>
 1.0  <Build 3>
  <Build 1>
ion 1.1  <Build 1>
  <Build 1>
  <Build 1>
  <Build 1>
  <Build 4>
1.1  <Build 1>
0  <Build 1>

Snort successfully validated the configuration!
Snort exiting
root@student:/home/student/Downloads# snort -A console -q -c /etc/snort/snort.conf -i enp0s3
06/21-13:50:57.344843  [**] [1:1000002:0] Test rule [**] [Priority: 0] {TCP} 31.3.245.133:80 -> 10.0.2.15:58
854
06/21-13:50:57.345877  [**] [1:1000002:0] Test rule [**] [Priority: 0] {TCP} 31.3.245.133:80 -> 10.0.2.15:58
854
06/21-13:50:57.434713  [**] [1:498:6] ATTACK-RESPONSES id check returned root [**] [Classification: Potentia
lly Bad Traffic] [Priority: 2] {TCP} 31.3.245.133:80 -> 10.0.2.15:58854
06/21-13:50:57.434713  [**] [1:1882:10] ATTACK-RESPONSES id check returned userid [**] [Classification: Pote
ntially Bad Traffic] [Priority: 2] {TCP} 31.3.245.133:80 -> 10.0.2.15:58854
06/21-13:50:57.434713  [**] [1:1000002:0] Test rule [**] [Priority: 0] {TCP} 31.3.245.133:80 -> 10.0.2.15:58
854
06/21-13:50:57.436783  [**] [1:1000002:0] Test rule [**] [Priority: 0] {TCP} 31.3.245.133:80 -> 10.0.2.15:58
854
06/21-13:50:57.528525  [**] [1:1000002:0] Test rule [**] [Priority: 0] {TCP} 31.3.245.133:80 -> 10.0.2.15:58
854
06/21-13:50:57.528543  [**] [1:1000002:0] Test rule [**] [Priority: 0] {TCP} 10.0.2.15:58854 -> 31.3.245.133
:80
06/21-13:50:59.434090  [**] [1:1000002:0] Test rule [**] [Priority: 0] {TCP} 52.10.115.149:443 -> 10.0.2.15:
41908
06/21-13:50:59.434750  [**] [1:1000002:0] Test rule [**] [Priority: 0] {TCP} 52.10.115.149:443 -> 10.0.2.15:
41908
```

Links for research further about URI's: https://www.w3.org/Addressing/URL/uri-spec.html & https://tools.ietf.org/html/rfc3986

4. Deliverable

Lab Activity can be submitted as either pdf or word document. Please include your answers and screenshots of relevant outputs.

References

https://tacticalflex.zendesk.com/hc/en-us/articles/360010598474-How-Snort-Network-Intrusion-Detection-System-Can-Successfully-Counter-Block-and-Detect-Malware

https://www.manpagez.com/man/8/snort/

https://ieeexplore.ieee.org/document/6914042

https://unixmen.com/install-snort-nids-ubuntu-15-04/

https://resources.infosecinstitute.com/snort-rules-workshop-part-one/#gref

https://paginas.fe.up.pt/~mgi98020/pgr/writing_snort_rules.htm

https://www.informit.com/articles/article.aspx?p=101171&seqNum=2

https://blog.snort.org/2011/09/flow-matters.html

https://www.ciscolive.com/c/dam/r/ciscolive/emea/docs/2019/pdf/BRKSEC-3352.pdf