Assignment Information			
Name:	Mario Morales	Assignment:	Project 1
Date Submitted:		Course Section:	
Course:	COSN 215		

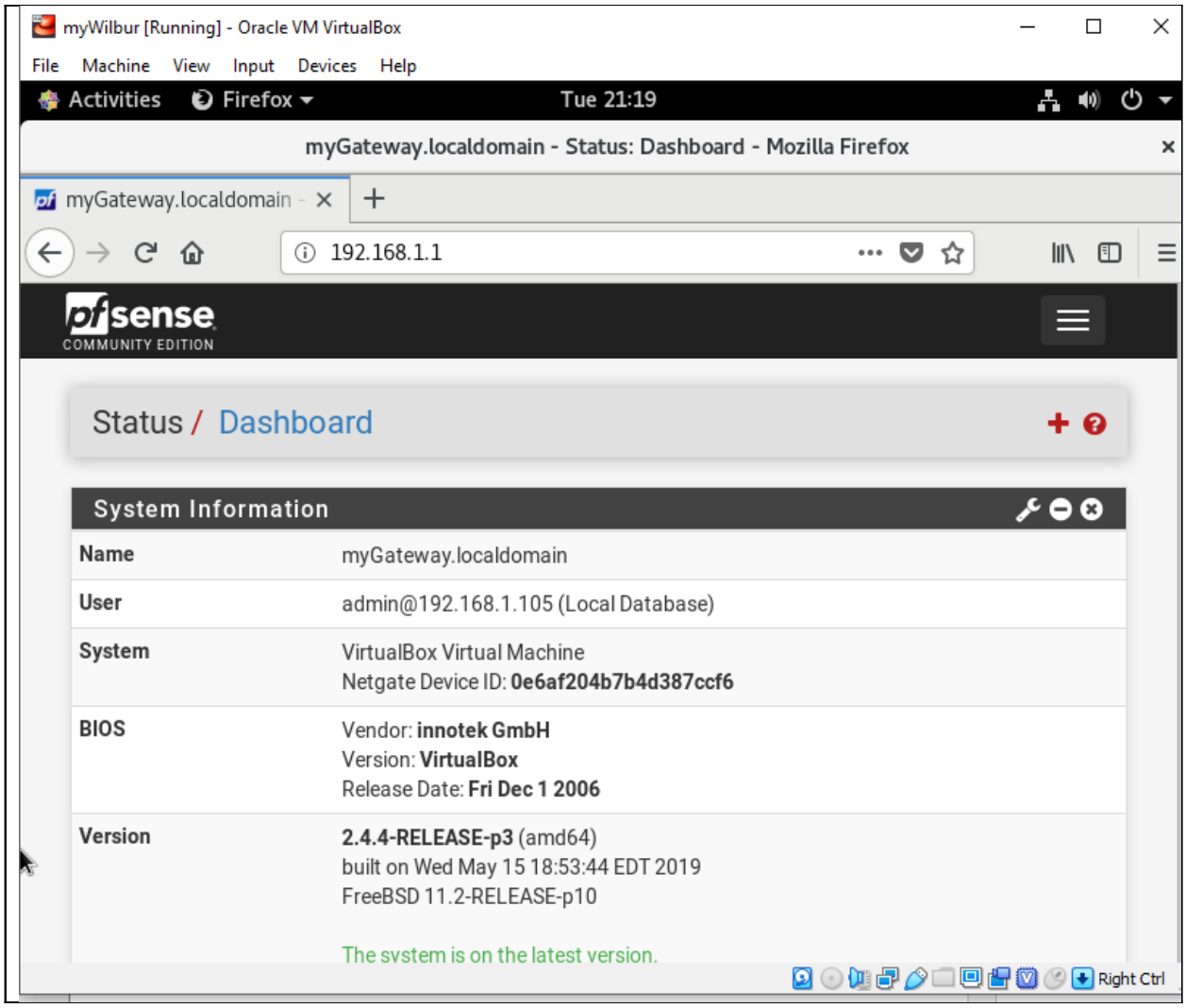
(IMPORTANT)

```
236  sudo nano /etc/snort/rules/myHousemy.rules
237  sudo snort -A console -c /etc/snort/snort.conf
238  sudo snort -T -c /etc/snort/snort.conf
239  history
```

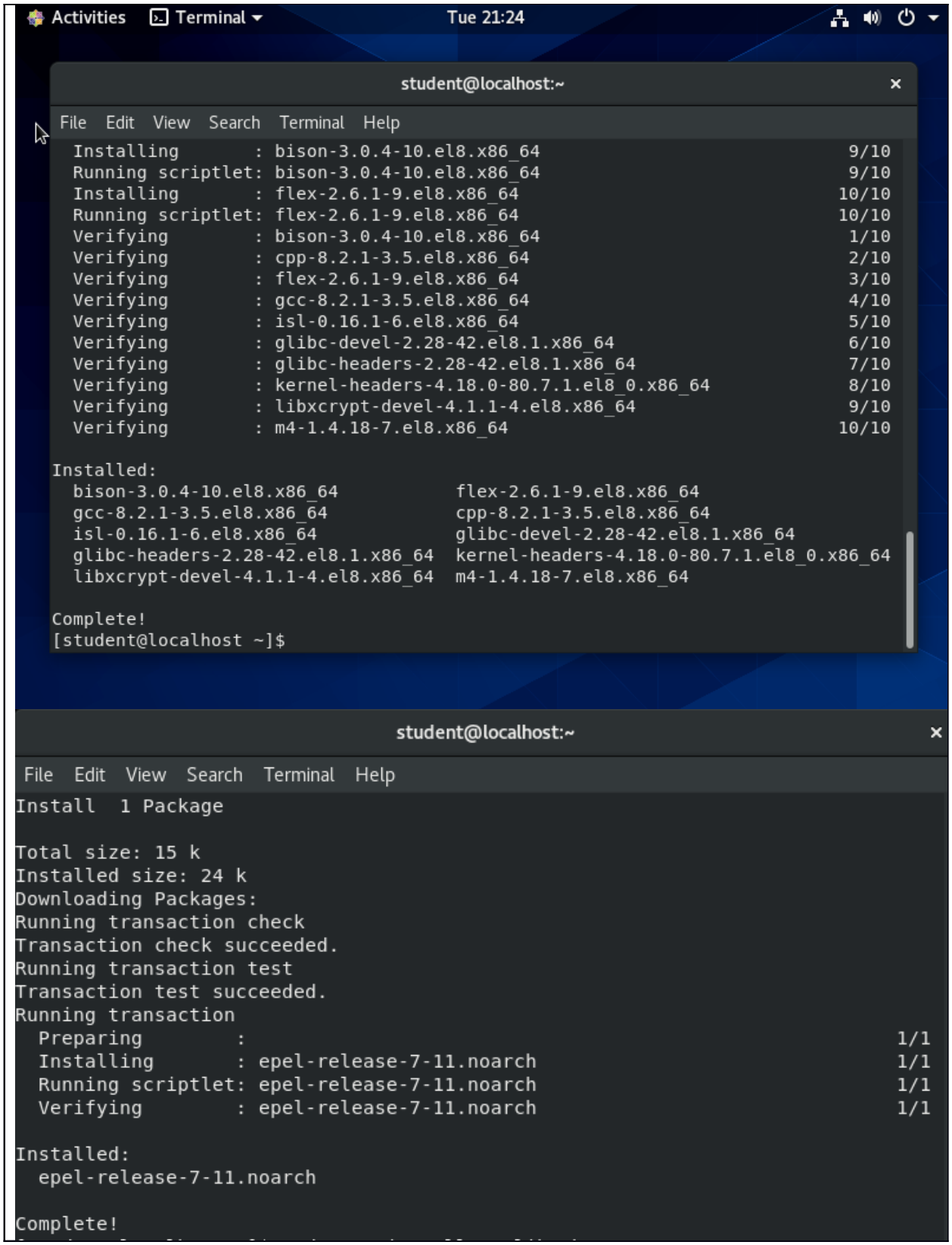
The purpose of this project is to secure the DMZ you've been building with an IDS and a vulnerability scan. Because of some of the limitations of our virtual environment we will be using 2 IDSs, first installed to a new machine on our DMZ and a second installed to myGateway

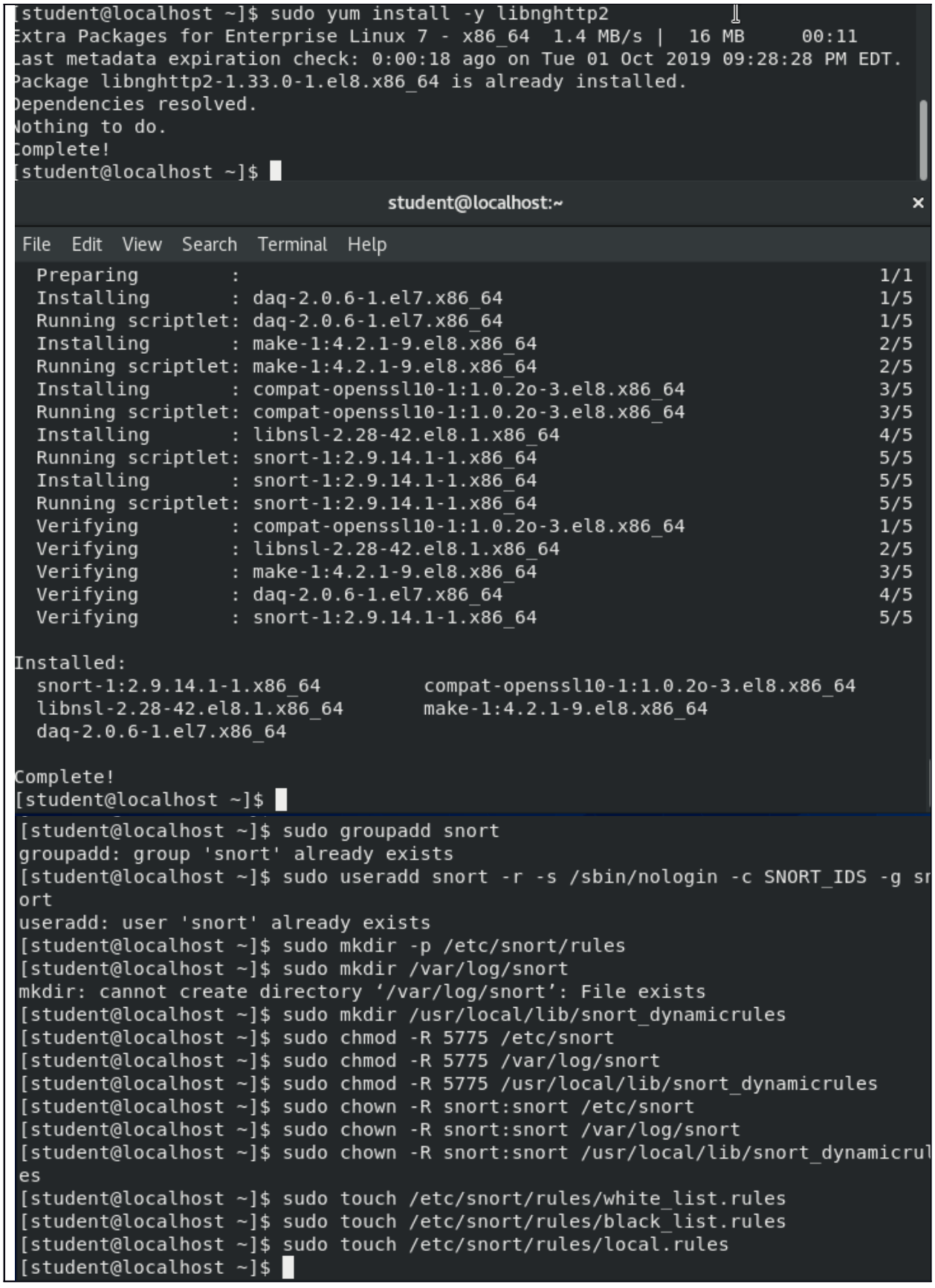
Phase 0

Install CentOS (name the machine **mySpiderHam**) in your *DMZ* network with the username/password **student/P@ssw0rd**. Configure your network interface to 192.168.2.2 and to connect through **myGateway**. Post a screenshot showing the finished install and networking:



Install the **snort** and post screenshots of each of the steps:


Activities
Terminal
Tue 21:24
student@localhost:~
File Edit View Search Terminal Help
Installing : bison-3.0.4-10.el8.x86_64 9/10
Running scriptlet: bison-3.0.4-10.el8.x86_64 9/10
Installing : flex-2.6.1-9.el8.x86_64 10/10
Running scriptlet: flex-2.6.1-9.el8.x86_64 10/10
Verifying : bison-3.0.4-10.el8.x86_64 1/10
Verifying : cpp-8.2.1-3.5.el8.x86_64 2/10
Verifying : flex-2.6.1-9.el8.x86_64 3/10
Verifying : gcc-8.2.1-3.5.el8.x86_64 4/10
Verifying : isl-0.16.1-6.el8.x86_64 5/10
Verifying : glibc-devel-2.28-42.el8.1.x86_64 6/10
Verifying : glibc-headers-2.28-42.el8.1.x86_64 7/10
Verifying : kernel-headers-4.18.0-80.7.1.el8_0.x86_64 8/10
Verifying : libxcrypt-devel-4.1.1-4.el8.x86_64 9/10
Verifying : m4-1.4.18-7.el8.x86_64 10/10
Installed:
bison-3.0.4-10.el8.x86_64 flex-2.6.1-9.el8.x86_64
gcc-8.2.1-3.5.el8.x86_64 cpp-8.2.1-3.5.el8.x86_64
isl-0.16.1-6.el8.x86_64 glibc-devel-2.28-42.el8.1.x86_64
glibc-headers-2.28-42.el8.1.x86_64 kernel-headers-4.18.0-80.7.1.el8_0.x86_64
libxcrypt-devel-4.1.1-4.el8.x86_64 m4-1.4.18-7.el8.x86_64
Complete!
[student@localhost ~]$
student@localhost:~
File Edit View Search Terminal Help
Install 1 Package
Total size: 15 k
Installed size: 24 k
Downloading Packages:
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing : 1/1
Installing : epel-release-7-11.noarch 1/1
Running scriptlet: epel-release-7-11.noarch 1/1
Verifying : epel-release-7-11.noarch 1/1
Installed:
epel-release-7-11.noarch
Complete!

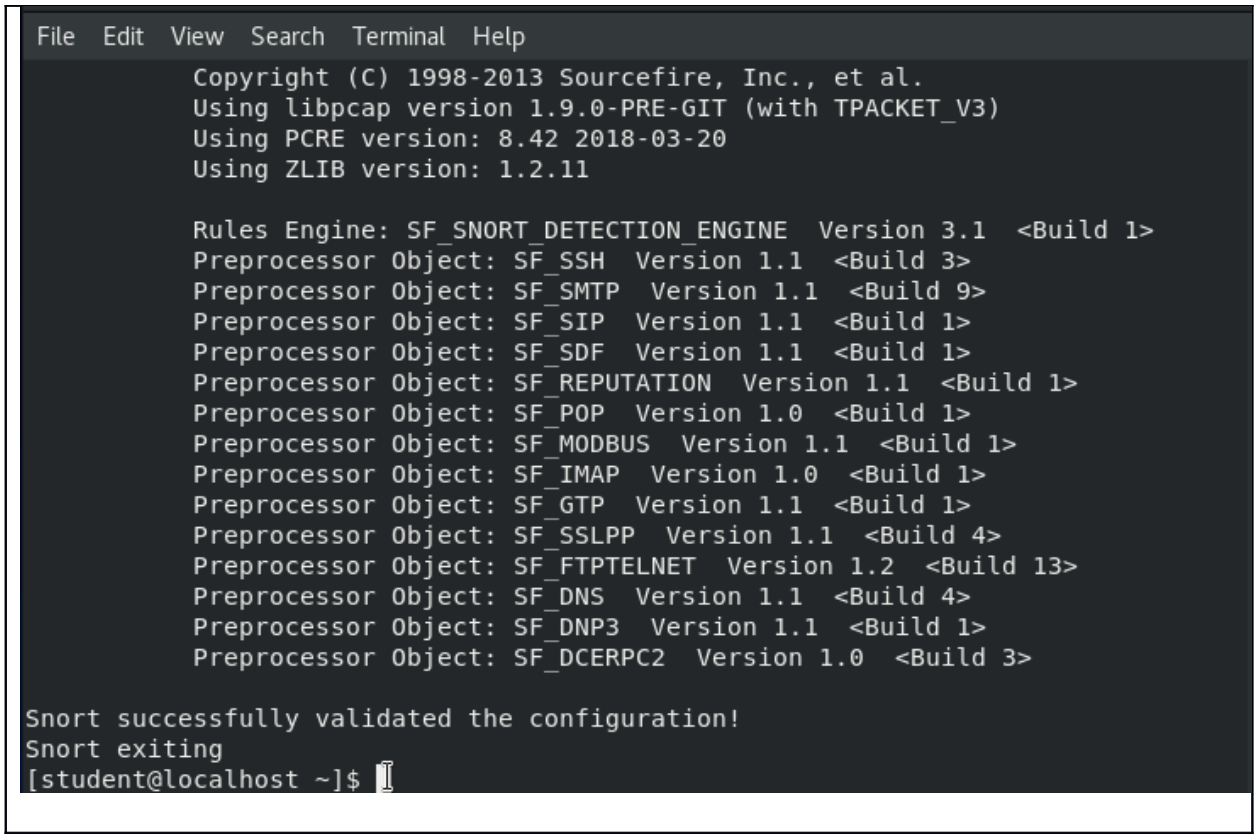

[student@localhost ~]$ sudo yum install -y libnghttp2
Extra Packages for Enterprise Linux 7 - x86_64 1.4 MB/s | 16 MB   00:11
Last metadata expiration check: 0:00:18 ago on Tue 01 Oct 2019 09:28:28 PM EDT.
Package libnghttp2-1.33.0-1.el8.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[student@localhost ~]$
student@localhost:~
File Edit View Search Terminal Help
Preparing : 1/1
Installing : daq-2.0.6-1.el7.x86_64 1/5
Running scriptlet: daq-2.0.6-1.el7.x86_64 1/5
Installing : make-1:4.2.1-9.el8.x86_64 2/5
Running scriptlet: make-1:4.2.1-9.el8.x86_64 2/5
Installing : compat-openssl10-1:1.0.2o-3.el8.x86_64 3/5
Running scriptlet: compat-openssl10-1:1.0.2o-3.el8.x86_64 3/5
Installing : libnsl-2.28-42.el8.1.x86_64 4/5
Running scriptlet: snort-1:2.9.14.1-1.x86_64 5/5
Installing : snort-1:2.9.14.1-1.x86_64 5/5
Running scriptlet: snort-1:2.9.14.1-1.x86_64 5/5
Verifying : compat-openssl10-1:1.0.2o-3.el8.x86_64 1/5
Verifying : libnsl-2.28-42.el8.1.x86_64 2/5
Verifying : make-1:4.2.1-9.el8.x86_64 3/5
Verifying : daq-2.0.6-1.el7.x86_64 4/5
Verifying : snort-1:2.9.14.1-1.x86_64 5/5
Installed:
snort-1:2.9.14.1-1.x86_64 compat-openssl10-1:1.0.2o-3.el8.x86_64
libnsl-2.28-42.el8.1.x86_64 make-1:4.2.1-9.el8.x86_64
daq-2.0.6-1.el7.x86_64
Complete!
[student@localhost ~]$
[student@localhost ~]$ sudo groupadd snort
groupadd: group 'snort' already exists
[student@localhost ~]$ sudo useradd snort -r -s /sbin/nologin -c SNORT_IDS -g sn
ort
useradd: user 'snort' already exists
[student@localhost ~]$ sudo mkdir -p /etc/snort/rules
[student@localhost ~]$ sudo mkdir /var/log/snort
mkdir: cannot create directory ‘/var/log/snort’: File exists
[student@localhost ~]$ sudo mkdir /usr/local/lib/snort_dynamicrules
[student@localhost ~]$ sudo chmod -R 5775 /etc/snort
[student@localhost ~]$ sudo chmod -R 5775 /var/log/snort
[student@localhost ~]$ sudo chmod -R 5775 /usr/local/lib/snort_dynamicrules
[student@localhost ~]$ sudo chown -R snort:snort /etc/snort
[student@localhost ~]$ sudo chown -R snort:snort /var/log/snort
[student@localhost ~]$ sudo chown -R snort:snort /usr/local/lib/snort_dynamicrul
es
[student@localhost ~]$ sudo touch /etc/snort/rules/white_list.rules
[student@localhost ~]$ sudo touch /etc/snort/rules/black_list.rules
[student@localhost ~]$ sudo touch /etc/snort/rules/local.rules
[student@localhost ~]$

```
_request&X-Amz-Date=20191002T020411Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host
&X-Amz-Signature=5db049111917c044942660f3df34c28861d1c621611e96d1db7f5f0fa19755f
9 [following]
--2019-10-01 22:04:11--  https://snort-org-site.s3.amazonaws.com/production/rele
ase_files/files/000/011/691/original/snortrules-snapshot-29120.tar.gz?X-Amz-Algo
rithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIXACIED2SPMSC7GA%2F20191002%2Fus-eas
t-1%2Fs3%2Faws4_request&X-Amz-Date=20191002T020411Z&X-Amz-Expires=3600&X-Amz-Sig
nedHeaders=host&X-Amz-Signature=5db049111917c044942660f3df34c28861d1c621611e96d1
db7f5f0fa19755f9
Resolving snort-org-site.s3.amazonaws.com (snort-org-site.s3.amazonaws.com)... 5
2.216.137.220
Connecting to snort-org-site.s3.amazonaws.com (snort-org-site.s3.amazonaws.com)|
52.216.137.220|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 95458428 (91M) [application/octet-stream]
Saving to: ‘snortrules-snapshot-29120.tar.gz?oinkcode=9c6ac3ffa2ad9c54a7bd84071f
b419570d85b703’

snortrules-snapshot 100%[===================>]  91.04M  2.64MB/s    in 38s

2019-10-01 22:04:49 (2.39 MB/s) - ‘snortrules-snapshot-29120.tar.gz?oinkcode=9c6
ac3ffa2ad9c54a7bd84071fb419570d85b703’ saved [95458428/95458428]

so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/server-mail.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/protocol-other.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/file-java.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/file-pdf.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/malware-other.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/file-other.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/protocol-snmp.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/exploit-kit.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/server-oracle.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/os-other.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/file-image.so
so_rules/precompiled/Alpine-3-10/x86-64/2.9.12.0/file-executable.so
etc/
etc/classification.config
etc/reference.config
etc/sid-msg.map
etc/snort.conf
etc/threshold.conf
etc/unicode.map
preproc_rules/
preproc_rules/decoder.rules
preproc_rules/preprocessor.rules
```



Explain the things you had to do to make the installation happen:

Download and install snort with YUM utility Confiuring snort to run in NIDS mode Setting up username & folder structure Using registered user rules Validating settings Testing configuration Running snort in the background

Take a minute and read the /etc/snort.conf file before continuing. Breath it in. Now give a brief (1-3 sentence) explanation of each section:

Set the network variables	In this section you can “set” variables that are used throughout the config file. These include networks, paths, and files.
Configure the decoder	Decoder and preprocessor rules allow one to enable and disable decoder and preprocessor events on a rule by rule basis
Configure the base detection engine	The detection engine is the meat of the IDS in Snort. The detection engine takes the data that comes from the preprocessor and its

	plug-ins, and that data is checked through a set of rules
Configure dynamic loaded libraries	Tells snort to load the dynamic engine shared library
Configure preprocessors	It reassembles packets into meaningful sessions for the Snort rules
Configure output plugins	The output modules are run when the alert or logging subsystems of Snort are called
Customize your rule set	Add custom rules
Customize preprocessor and decode rule set	Add custom preprocessor and decode rule set
Customize shared object rule set	Add custom shared object rules

Phase 1

Will we be running snort as inline or passive? As an IDS or IPS? Explain:

Passive, IDS. Passive so it runs in the background. IDS because I want to detect threats.

From **myPenTestDMZ** start ping the ip address of **mySpiderHam** and leave that running.

Create a custom rule list called *myHousemy.rules* on **mySpiderHam.** Create a rule to alert on ping traffic anywhere on the network. Post a screenshot of the rule and a successful config test here:

```
alert icmp any any -> $HOME_NET any (msg:"ICMP test"; sid:1000002; rev:002;)
```

```
File  Edit  View  Search  Terminal  Help
           Using PCRE version: 8.42 2018-03-20
           Using ZLIB version: 1.2.11

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 3.1  <Build 1>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>

Snort successfully validated the configuration!
Snort exiting
```

Post a screenshot of the snort console showing the *ping* traffic from **myPenTestDMZ.**

```
Commencing packet processing (pid=5813)
10/08-23:28:59.391272  [**] [1:1000002:2] ICMP test [**] [Priority: 0] {ICMP} 1
2.168.1.104 -> 192.168.1.105
10/08-23:28:59.391317  [**] [1:1000002:2] ICMP test [**] [Priority: 0] {ICMP} 1
2.168.1.105 -> 192.168.1.104
10/08-23:29:00.399645  [**] [1:1000002:2] ICMP test [**] [Priority: 0] {ICMP} 1
2.168.1.104 -> 192.168.1.105
10/08-23:29:00.399681  [**] [1:1000002:2] ICMP test [**] [Priority: 0] {ICMP} 1
2.168.1.105 -> 192.168.1.104
10/08-23:29:01.424051  [**] [1:1000002:2] ICMP test [**] [Priority: 0] {ICMP} 1
2.168.1.104 -> 192.168.1.105
10/08-23:29:01.424086  [**] [1:1000002:2] ICMP test [**] [Priority: 0] {ICMP} 1
2.168.1.105 -> 192.168.1.104
10/08-23:29:02.447684  [**] [1:1000002:2] ICMP test [**] [Priority: 0] {ICMP} 1
2.168.1.104 -> 192.168.1.105
10/08-23:29:02.447719  [**] [1:1000002:2] ICMP test [**] [Priority: 0] {ICMP} 1
2.168.1.105 -> 192.168.1.104
```

Now ssh from **myPenTestDMZ** to **mySpiderHam**. Post a successful screenshot here:

```
root@myPenTestDMZ:~# ssh student@192.168.1.105
student@192.168.1.105's password:
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Tue Oct  1 23:14:22 2019
[student@localhost ~]$
```

Create a rule that would identify ssh traffic into **mySpiderHam** and block it**.** Test and start it up, show the rule and test here:

```
reject tcp any any -> 192.168.1.105 22 (msg:"SSH test"; sid:1000001; rev:001;)
```

```
File  Edit  View  Search  Terminal  Help
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.9.0-PRE-GIT (with TPACKET_V3)
           Using PCRE version: 8.42 2018-03-20
           Using ZLIB version: 1.2.11

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 3.1  <Build 1>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>

Snort successfully validated the configuration!
Snort exiting
[student@localhost snort]$
```

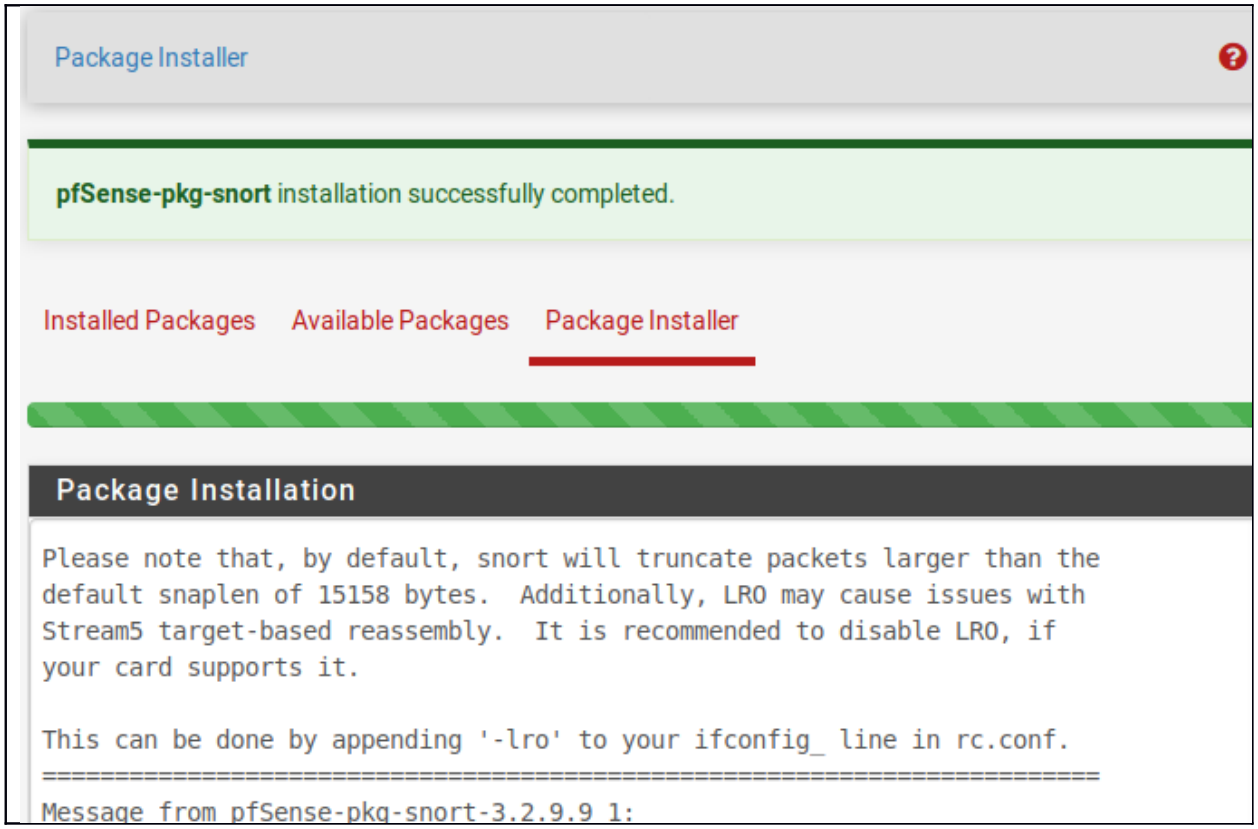
Now show that the sh from **myPenTestDMZ** to **mySpiderHam** is blocked. Also post what the console alert displays:

```
root@myPenTestDMZ:~# ssh student@192.168.1.105
Connection reset by 192.168.1.105 port 22
root@myPenTestDMZ:~# ssh student@192.168.1.105
student@192.168.1.105's password:
```

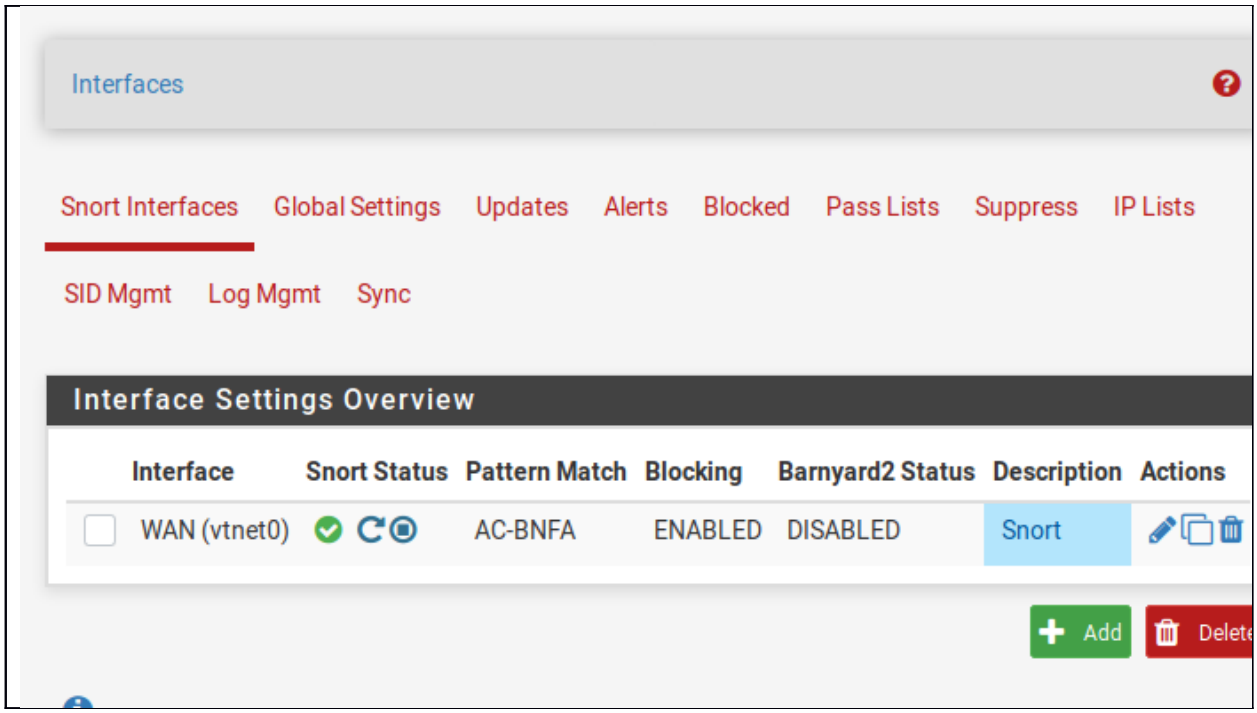
```
168.1.104:51264 -> 192.168.1.105:22
10/08-23:24:36.323577  [**] [1:1000001:1] SSH test [**] [Priority: 0] {TCP} 19
168.1.104:51264 -> 192.168.1.105:22
10/08-23:24:36.324041  [**] [1:1000001:1] SSH test [**] [Priority: 0] {TCP} 19
168.1.104:51264 -> 192.168.1.105:22
10/08-23:24:36.376112  [**] [1:1000001:1] SSH test [**] [Priority: 0] {TCP} 19
168.1.104:51264 -> 192.168.1.105:22
```

Phase 2

Now you're going to install snort on **myGateway** and test it there with a standard rule configuration.
Install snort on **myGateway** and paste a screenshot here:



Install the full community rule set and enable it on the WAN for **myGateway** and paste a screenshot here:



Create a new Kali install called **myPenTestWAN** and use it to scan **myGateway**. What are you picking up? Show/discuss:

```
root@kali:~# nmap 10.201.112.109
Starting Nmap 7.80 ( https://nmap.org ) at 2019-10-15 18:42 PDT
Nmap scan report for 10.201.112.109
Host is up (0.0012s latency).
Not shown: 997 filtered ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 4.87 seconds
root@kali:~#
```

Picking up ports that are open on the network. 22/80/443 that were written on the pfsense firewall, every other port is “filtered” which means nmap cannot access them due to being blocked by snort.