CSci 4271W (011 and 012 Sections) Lab Instructions

Lab 9 **March 31st, 2025**

Ground Rules. You may choose to complete this lab in a group of up to three students. Before you leave the lab, **make sure you have submitted to Gradescope, you included all group members on the submission, and the autograder found all required files!**

1 Oh no, you are a scary Snort!

In today's lab, we'll see a widely-used intrusion detection system, Snort. Snort is open-source and has a large community of users that contribute to a huge list of "community rules" that can match known attacks on network applications. It is used by many large enterprises including the UMN network security team.

(This lab was adapted from a Snort lab offered by the Infosec Institute)

2 Installing snort

For today's lab, you'll want to have multiple terminals open to your VM and on the CSELabs machine you're using, which we will assume is csel-wb28-LL (for a value of LL between 01 and 27). (Even if you're not physically in the lab, you need to use terminals SSHed to a lab machine because both the instructions and the autograder are based on their IP addresses.) We'll be doing some editing of text files on the VM, so unless you have a terminal-based text editor you know and like (e.g., `emacs -nw`, `vim`, or `nano`) you might want to use X forwarding when you ssh to your VM, e.g. `ssh -X student@csel-xsme-s25-csci4271-NNN`. Once you've logged in to your VM, you can install snort in your VM by running the following command:

```
$ sudo apt-get install snort
```

This will download about several files and may take a few minutes to finish. While it's installing, you may be asked which interface to configure snort to use: if so, the textbox will initially contain `eth0`, and you should replace it with `ens18`; you'll be asked what the "home network" of your VM is, and the text box will initially be populated with `192.168.0.0/16`; You should replace this with `10.32.102.0/24`.

3 Running snort

Let's start by verifying that snort installed correctly. In your VM terminal, type:

```
$ snort -V
```

You should see the message:

```
   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.20 GRE (Build 82)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2022 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.10.4 (with TPACKET_V3)
           Using PCRE version: 8.39 2016-06-14
           Using ZLIB version: 1.3
```

Now, because the `apt` installation set up files to be used on a production server, and we just want to experiment with a minimal configuration, we'll grab a reduced configuration file:

```
$ git clone https://github.umn.edu/badlycoded/lab9.git
```

Now we're ready to run snort, although we haven't configured it to use any rules yet. To verify this, run the following command:

```
$ sudo snort -T -i ens18 -c ~/lab9/snort.conf
```

Here the option `-T` tells snort to (T)est the configuration file, the option `-i` tells snort what network (i)nterface to listen on, and `-c` tells snort what (c)onfig file to use (here it's the `snort.conf` file in your VM home directory). Running the command will result in a lot of output, but if you scroll back up in the terminal window, you should see a section that looks like this:

```
++++++++++++++++++++++++++++++++++++++++++++++++++
Initializing rule chains...
0 Snort rules read
    0 detection rules
    0 decoder rules
    0 preprocessor rules
0 Option Chains linked into 0 Chain Headers
++++++++++++++++++++++++++++++++++++++++++++++++++
```

Let's create our first simple test rule. This rule will generate an alert whenever Snort detects an ICMP Echo request (ping) or Echo reply message. Open snort's `local.rules` file in a text editor as root with the following command:

```
$ sudo gedit /etc/snort/rules/local.rules
```

You should see that the file has some comment lines (beginning with `#`) but no rules. Add the following rule after the comments (no line breaks):

```
alert icmp any any -> $HOME_NET any (msg:"ICMP test"; sid:1000001; rev:1; classtype:icmp-event;)
```

And save the file. What is this rule telling snort?

- *Rule Header*: The portion before the parens is the header:

  - the `alert` keyword is the rule *action*. Snort will generate an alert when the rule is matched.

  - `icmp` specifies what protocol to match
  - the first `any` means that the rule will match any source IP address
  - the second `any` means that the rule will match any source port
  - `->` is the direction (from source to local network)
  - `$HOME_NET` means that the destination IP is in the local network we set in the configuration to be the network containing the VMs.
  - the third `any` means the rule will match any destination port.

- *Rule Options*: The next part specifies additional options about the rule:

  - `msg:"ICMP test"` - the message to print with the alert
  - `sid:1000001` - Snort rule ID (aka "signature ID"). All numbers $< 1{,}000{,}000$ are reserved, so we are starting with 1000001 (you may use any number, as long as it's greater than 1,000,000).
  - `rev:1` - Revision number. This option allows for easier rule maintenance.
  - `classtype:icmp-event` - Categorizes the rule as an "icmp-event", one of the predefined Snort categories. This option helps with rule organization.

Now if we run `sudo snort -T -i ens18 -c ~/lab9/snort.conf` again and scroll up through the output, we should see:

```
++++++++++++++++++++++++++++++++++++++++++++++++++
Initializing rule chains...
1 Snort rules read
    1 detection rules
    0 decoder rules
    0 preprocessor rules
1 Option Chains linked into 1 Chain Headers
++++++++++++++++++++++++++++++++++++++++++++++++++
```

Now, let's start Snort in IDS mode and tell it to display alerts to the console:

```
$ sudo snort -A console -q -c ~/lab9/snort.conf -i ens18
```

Again, we are pointing Snort to the configuration file it should use (`-c`) and specifying the interface (`-i ens18`). The `-A console` option prints alerts to standard output, and `-q` is for "quiet" mode (not showing banner and status report). You shouldn't see any output when you enter the command because Snort hasn't detected any activity specified in the rule we wrote. Let's generate some activity and see if our rule is working.

In your CSELabs terminal, start pinging your VM with the command (substitute the appropriate value for `NNN`):

```
$ ping csel-xsme-s25-csci4271-NNN
```

Let it run for 5 or 6 pings and then stop it with <Ctrl-C>. Back in your VM terminal you should see the alerts generated by these requests:

```
03/31-13:26:13.979397  [**] [1:1000001:1] ICMP test [**] [Classification: Generic ICMP event]
↪  [Priority: 3] {ICMP} 134.84.56.103 -> 10.32.102.63
03/31-13:26:14.980887  [**] [1:1000001:1] ICMP test [**] [Classification: Generic ICMP event]
↪  [Priority: 3] {ICMP} 134.84.56.103 -> 10.32.102.63
03/31-13:26:16.005248  [**] [1:1000001:1] ICMP test [**] [Classification: Generic ICMP event]
↪  [Priority: 3] {ICMP} 134.84.56.103 -> 10.32.102.63
03/31-13:26:17.029297  [**] [1:1000001:1] ICMP test [**] [Classification: Generic ICMP event]
↪  [Priority: 3] {ICMP} 134.84.56.103 -> 10.32.102.63
03/31-13:26:18.053284  [**] [1:1000001:1] ICMP test [**] [Classification: Generic ICMP event]
↪  [Priority: 3] {ICMP} 134.84.56.103 -> 10.32.102.63
03/31-13:26:19.077246  [**] [1:1000001:1] ICMP test [**] [Classification: Generic ICMP event]
↪  [Priority: 3] {ICMP} 134.84.56.103 -> 10.32.102.63
03/31-13:26:20.101454  [**] [1:1000001:1] ICMP test [**] [Classification: Generic ICMP event]
↪  [Priority: 3] {ICMP} 134.84.56.103 -> 10.32.102.63
```

(The addresses and dates you see will be slightly different.) You can see that the alerts include the `msg` we specified (`ICMP test`), along with the SID, event type, and the IP address triggering the event (`134.84.56.XX`). Looks like our test rule worked, so we can stop snort with <Ctrl-C> to return to the prompt.

Now let's write another rule, this time, a bit more specific. Open our local.rules file in a text editor:

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

First, let's comment out our first rule. Put a hash sign (#) in front of it. On a new line, write the following rule:

```
alert tcp 134.84.56.0/24 any -> $HOME_NET 80 (msg:"HTTP connection attempt"; sid:1000002; rev:1;)
```

Here we changed the protocol to TCP, used a more specific source IP (the subnet containing the lab machines), set the destination port number to 80 (default port for HTTP connections), and changed the alert message text. Save and close the file. Now let's run Snort in IDS mode again, but this time, we are going to add one more option, as follows:

```
$ sudo snort -A console -q -c ~/lab9/snort.conf -i ens18 -K ascii
```

We are telling Snort to log generated alerts in the ASCII format rather than the default pcap. Once Snort is running (again, you won't see any output right away), we need to set up some traffic that will trigger the rule again. This will be a little more involved than before. In another terminal connected to your VM, let's use python to start a very simple web server:

```
$ sudo python3 -m http.server 80
```

Note that we're running as root so we can use the "privileged" port 80; we will not want to keep this server running after the lab. Now that we have a web server running, we can send it some traffic. Switch to a CSELabs terminal window, and use wget to request the "index" page from this server:

```
$ wget http://csel-xsme-s25-csci4271-NNN/
```

(Since there's no `index.html` file the web server will generate a directory listing.)

Switch back to the window running snort, and we should see several alerts:

```
03/31-13:29:07.625122  [**] [1:1000002:1] HTTP connection attempt [**] [Priority: 0] {TCP}
↪  134.84.56.103:60322 -> 10.32.102.62:80
03/31-13:29:07.625831  [**] [1:1000002:1] HTTP connection attempt [**] [Priority: 0] {TCP}
↪  134.84.56.103:60322 -> 10.32.102.62:80
03/31-13:29:07.625991  [**] [1:1000002:1] HTTP connection attempt [**] [Priority: 0] {TCP}
↪  134.84.56.103:60322 -> 10.32.102.62:80
03/31-13:29:07.626860  [**] [1:1000002:1] HTTP connection attempt [**] [Priority: 0] {TCP}
↪  134.84.56.103:60322 -> 10.32.102.62:80
03/31-13:29:07.627397  [**] [1:1000002:1] HTTP connection attempt [**] [Priority: 0] {TCP}
↪  134.84.56.103:60322 -> 10.32.102.62:80
```

Hit <`Ctrl-C`> to stop snort, and let's take a look at snort's log directory:

```
$ sudo ls /var/log/snort
```

You should see something like:

```
134.84.56.103   snort.log.1726792494
```

The `snort.log.*` file (you may have more than one if you generated more than one alert-generating activity earlier) is the .pcap log file. It cannot be read with a text editor, but we could open it in Wireshark. The IP address that you see (yours will be different) is the source IP for the alert we just saw for our HTTP rule. It's a directory. Let's see what's inside:

```
$ sudo ls /var/log/snort/134.84.56.103
TCP:60322-80
```

We can see that there's a file there named after the port numbers (60322 and 80 in my case) and protocol (`TCP`) involved in the activity. Your directory will have a different file (`TCP:XXXXX-80`) You can view the file with `less`:

```
$ sudo less /var/log/snort/134.84.56.ZZZ/TCP:XXXXX-80
[**] HTTP connection attempt [**]
03/31-13:29:07.625122 134.84.56.103:60322 -> 10.32.102.62:80
TCP TTL:62 TOS:0x0 ID:54967 IpLen:20 DgmLen:60 DF
******S* Seq: 0x8C64F6AA  Ack: 0x0  Win: 0xFAF0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 3989547270 0 NOP WS: 7
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

We get the same information as we saw in the console output with some additional details about the packets: there will be one "entry" per packet in the flow. You can see the sequence of packets sent to your VM, e.g. the first packet has a SYN flag to open the connection, the next has ACK and PUSH flags because it includes the request from the CSElabs machine, and the final packet will have a FIN flag to close the connection. Let's copy this file to someplace more convenient for pulling off of your VM:

```
$ sudo cp /var/log/snort/134.84.56.ZZZ/TCP:XXXXX-80 /home/student/snort80.log
$ sudo chown student:student /home/student/snort80.log
```

Now you can use `scp` to copy the `snort80.log` file off of your VM for submission on gradescope.

Cleanup note: remember to stop the python http.server that you started in the other VM terminal. You can stop the process with <`Ctrl-C`>.

3.1 All done!

Once you've captured/alerted an HTTP connection attempt and inspected the capture file, you're done with Lab 9! Use `scp` to copy the `snort80.log` file off of your VM, so you can submit it to the Lab 9 assignment on Gradescope.

Once you've submitted the file, the autograder will test to make sure the proper file was submitted, check that it includes the right information, and notify you if anything went wrong, within a few minutes.

Congratulations, you've finished Lab 9!