Introduction

In this lab, you will practice configuration of the snort intrusion detection system (IDS).

This document describes the tasks you will do in the lab and must not be viewed as a manual or tutorial (although rather detailed information is provided). You are expected to use the slides from the lecture, to read the documents available under Links in Its Learning and to make full use of the search engines on the web. See also *Deliverables* section at the end of this document.

The VirtualBox appliance from *Lab 1: Linux networking and firewalls* will be reused here. You should reboot any VMs you have started, so that iptables configuration is reset the original Default ACCEPT policy. The ACCEPT policy will simplify debugging the lab.

If you have made persistent changes to the iptables rules (changes that cannot be undone by a reboot) you can restore the VM to the state saved in the *Complete* snapshot. However, if you restore the VM, make sure to backup your changes (in particular the firewall.sh script) since **all** changes to the VM will be gone. Alternatively, VirtualBox will ask you if you want to take a snapshot of the current state before restoring the *Complete* snapshot. This is also an acceptable solution to backup your changes.

Metasploit

Metasploit is penetration testing software that is used by white-hat hackers in ethical hacking attempts to verify the security of IT systems[1]. It contains a large library with known exploits and attack implementations that can be easily deployed.

You will install the Metasploit Framework edition (free) on Server B. To do, that issue the following command[2]:

```
curl https://raw.githubusercontent.com/rapid7/metasploit-
omnibus/master/config/templates/metasploit-framework-wrappers/msfupdate.erb
> msfinstall && \
  chmod 755 msfinstall && \
  ./msfinstall
```

Enter the command msfconsole to start the Metasploit console. If you are asked if you like to use and setup a new database, then answer “yes”. To exit from the msfconsole enter “quit” at the msf prompt.

[1] Although the product is advertised as aiming towards improving security, there is nothing preventing black-hat users from using it in malicious ways. Always obtain written permission for carrying out cyber-attacks against systems other than your own.

[2] If unattended upgrades are being installed while you enter the command, you will get an error about being unable to get the repository lock. If that happens, wait a few minutes until the upgrades complete and then re-paste the command.

An in-depth tutorial for Metasploit is available at https://www.offensive-security.com/metasploit-unleashed/. Read the sections I*ntroduction*, *Metasploit fundamentals* and *Information Gathering* to get familiar with Metasploit.

Snort

Snort is an open-source intrusion detection system (IDS). It has a special inline mode that turns it into a full-blown intrusion prevention system (IPS). Snort comes with a large library of rules used for detecting attacks, but also allows users to expand the library with their own rules. In this lab, you will not use the built-in library, but instead write your own rules.

To install snort on server A enter the following command:

```
apt install snort
```

When asked what interfaces Snort should listen on, enter the name of the interface using IP address 192.168.70.5 as shown in Figure 1. Normally, it is *enp0s8*, but you should double-check.

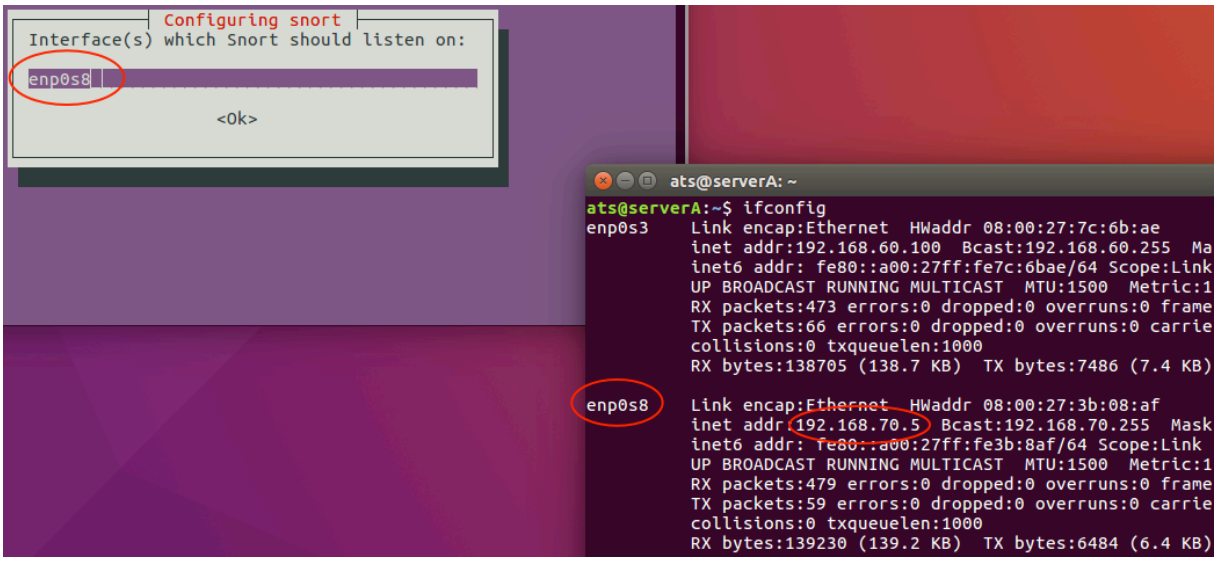

Figure 1: Installing snort

You will then be asked to enter the address range for the local network. Enter, 192.168.70.0/24 as shown in Figure 2. Accept the default choice for following questions.

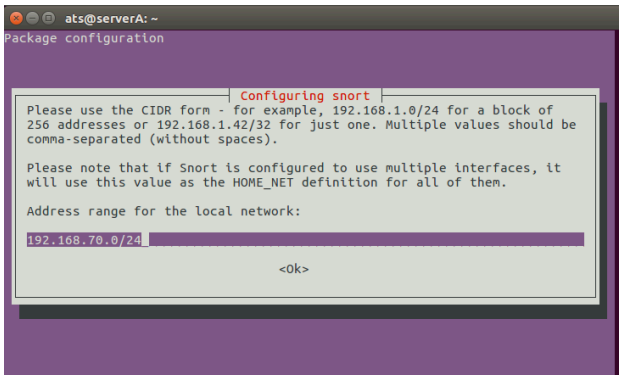

Figure 2: Range for local network

The manual for Snort is available at from https://www.snort.org/#documents in PDF and HTML formats. Read the first three chapters ("Snort Overview", "Configuring Snort", and "Writing Snort Rules") to become familiar with the operation. Focus particularly on Chapter 3 "Writing Snort Rules".

Environment

Only server A and server B are required for this lab. Both nodes will make use of the shared network 192.168.70.0/24. Just as in Lab 2, Server A's interface is assigned IP address 192.168.70.5, while server B's IP address is 192.168.70.6. Client A and Client B should be turned off to avoid any interference.

Server B will play the role of the malicious node that is doing reconnaissance or is actively attacking Server A with the help of Metasploit. You will run the Snort IDS system on Server A to detect reconnaissance attempts and incoming attacks.

Make sure that all host-only interfaces for Server A and Server B have promiscuous mode set to "Allow All", as shown in Figure 3. If the promiscuous mode is set to something else, you must first turn off the VM before being allowed to apply the change.

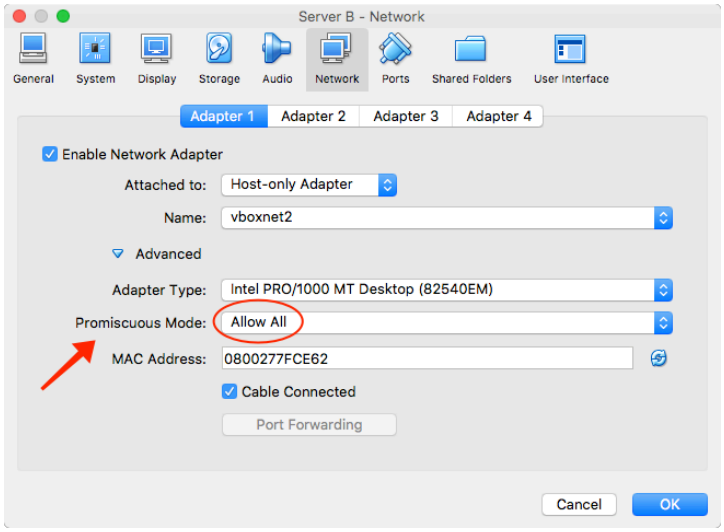

Figure 3: Promiscuous mode

Task 1: Check connectivity

Start Wireshark on Server A and begin capturing packets from interface assigned IP address 192.168.70.5. Ping Server A from Server B. Do you see the ICMP echo and reply packets?

If you answered "yes" to the question in Task 1, it means you have connectivity between Server A and Server B across the 192.168.70.0/24 network. Otherwise, you must diagnose the connection problem before proceeding with the next task.

In the next task, you will write your first Snort rule for detecting incoming pings (*i.e.*, ICMP Echo messages) from Server B to Server A. The rules are written to the file `/etc/snort/rules/local.rules` on Server A (where snort was installed).

Snort rules are of the form

```
<action> <proto> <srcIP> <srcPort> <dir> <dstIP> <dstPort> [ruleOptions]
```

The `<action>` field denotes what snort must do if the rule matches observed network traffic. For example, snort can log, alert, or drop the packet. In this lab, `<action>` will always be set `alert`.

The TCP/IP protocol that the rule must match is defined by the `<proto>` field. Legit options are `TCP`, `UDP`, `ICMP` and `IP`.

The field `<srcIP>` matches on the source IP address of the captured packet. The value can be a distinct IP address (e.g., 192.168.1.10), a subnet mask (e.g., 192.168.1.0/24) or the keyword `any`, which denotes that any IP address would match. Similarly, the `<srcPort>` field matches the source port of the captured packet. The value can be a distinct number (e.g., 21), a port range (e.g., 1:1024), or the keyword any, which denotes that any port number would match. Both `<srcIP>` and `<srcPort>` can be be negated through the use of the exclamation character (`!`). For example, if `<srcPort>` is set to `!1000:2000`, it will match any port number in the range `1-999` and `2001-65535`, but will not match the range `1000-2000`.

Same concepts apply to the fields `<dstIP>` and `<dstPort>` that match on destination IP address and destination port number, respectively, in the captured packet.

The field `<dir>` specifies unidirectional or bi-directional traffic. Unidirectional traffic, denoted by `->`, matches only traffic going from `srcIP` and `srcPort` towards `dstIP` and `dstPort`. The bi-directional operator, denoted by `<->`, matches the same traffic as the unidirectional operator as well as traffic going in the opposite direction, from `dstIP` and `dstPort` to `srcIP` and `srcPort`.

The `[ruleOptions]` is an optional part of the rules. If any options are present, they will be enclosed in the round brackets `()`. Each option consists of option name and option value separated by the colon character. Multiple options are separated by semicolon. At the very least, a message and a Snort rule id (sid) should accompany an alert, such as `(msg:"Packet matched"; sid:2000001)`. *The sid value must be unique for each rule*. You must choose values greater than 1000000 to avoid collisions with sid numbers allocated to the rules in the official snort library.

For additional details, see the Snort manual mentioned in *Snort* section.

Task 2: Detect incoming pings

Begin by editing the file `/etc/snort/snort.conf` on Server A. Use the pound character # to comment out the line `include $RULE_PATH/icmp-info.rules`

```
#include $RULE_PATH/icmp-info.rules
```

This will prevent the ICMP notification rules shipped with snort to be load so that you can write your own rules. Otherwise, you will see mixed output from your rules as well as the library rules.

Add the following rule to the file `/etc/snort/rules/local.rules` on Server A.

```
alert icmp 192.168.70.6 192.168.70.5 (msg:"ICMP message detected";
sid:2000001)
```

Note that the rule above wraps around the edge of the paper here due to space constraints. In the file, it must be entered on a single line (although the text editor you use may wrap the line also).

Save the file and start snort with the following command:

```
sudo snort -i enp0s8 -A console -c /etc/snort/snort.conf
```

where `enp0s8` is the interface assigned IP address 192.168.70.6. The directive `-A console` tells snort to print the alerts on screen and `-c /etc/snort/snort.conf` specifies the configuration file to be used.

Use a terminal on Server B to ping the address 192.168.70.6. Do you see matching alerts from snort on Server A? If yes, you have just written your first successful rule for snort. Include screenshots to prove it.

In the next task, you will write a snort rule on your own. You will also begin using the Metasploit framework to perform a reconnaissance activity.

Start Metasploit framework on a terminal on Server B by entering the command

```
sudo msfconsole
```

It may take a few seconds for it to complete loading the libraries, but eventually you will get a `msf >` prompt where you can enter commands.

The first command you should be aware of is `help`. If you enter help with no argument you will see a list with all available commands. Enter `help <cmd>`, where `<cmd>` is the name of a command to see help for that command. A useful feature is tab completion – enter the first letters of keyword and press the TAB key twice to let Metasploit show you all keywords beginning with that later. The tab completion can also reveal the available parameters for a

command. For example, type `show` and press TAB twice. Choose a parameter from the list shown, type the first letters of its name and press TAB twice to complete.

Snort comes with a large number of modules that implement various intrusion methods. The command `use <mod>` allows you to interact with a module specified by the path `<mod>`. Use tab completion to see what modules are available.

Once a module is selected you can view its description using the command `info`. The information displayed includes the list of variables used for configuring the module. Some of the variables are optional, but those that are mandatory must be set before the module can be used.

The syntax for setting a variable is `set <name> <value>`, where `<name>` is the variable name and `<value>` is the assigned value. The value can be a string, a number or a range (e.g., 1-1024). The values set this way have local scope – they exist only for the current module. The command `setg <name> <value>` allows you to set/define global variables.

After you are done with a module you can use the command `back` to return to the main menu. Use the command `exit` to quit Metasploit.

For more help on Metasploit see the tutorial at [https://www.offensive-security.com/metasploit-unleashed/.](https://www.offensive-security.com/metasploit-unleashed/)

Task 3: Detect TCP port scanning

Select the SYN port scanning in Metasploit by entering the following command:

```
use auxiliary/scanner/portscan/syn
```

Then, use the info command to read the module documentation. Before running the module, you must set the IP address of the remote host to be scanned (server A) and the server B's interface over which active probing will occur (i.e., the interface assigned the IP address 192.168.70.6 – typically `enp0s8`). You should also change the port range to 1-500 in order to reduce the amount of time spent in scanning.

```
set RHOSTS 192.168.70.5
set INTERFACE enp0s8
set PORTS 1-500
```

Start Wireshark on interface enp0s8 on server A to capture the scanning traffic. Open the menu `Edit->Preferences->Protocols->TCP`. Uncheck the *Relative sequence numbers* option to make Wireshark display the actual sequence numbers.

Return to Metasploit and enter the command run. You should be able to see in Wireshark pairs of TCP packets – a SYN request from Server B to Server A and a reply (typically RST/ACK) going in the opposite direction.

After the port scanning is complete Metaploit will show the list of all open ports in the range entered. These are ports used by an application or service that awaits incoming connections. Describe which open ports you have found on Server A and to what services they belong. You can find a name to port number mapping in the file `/etc/services`.

Examine the TCP header in the traffic captured with Wireshark and determine what are the differences between the packet pairs for open ports and those for closed ports. Based on this analysis explain how you think this type of port scanning works.

You will now write a snort rule to detect this form of port scanning. To do that, you must first identify a unique element in the packets used for scanning – one that sets them apart from benign traffic. It is useful to compare benign packets to the scanning packets to identify this element. We'll use telnet[3] and ssh for this purpose.

Open a terminal on Server B and enter the following command while capturing traffic on Server A:

```
telnet 192.168.70.5
```

By default, telnet connects to port 23. There is no service listening on that port on Server A and therefore the command returns immediately. You can also connect to the open ports found earlier by specifying the port number on the command line

```
telnet 192.168.70.5 <port>
```

where `<port>` is the port number.

Compare the TCP headers from the telnet packets going towards Server A to the TCP headers from the scanning packets going in the same direction. Look at the acknowledgement number and the flags fields in particular. Compare also the TTL fields in the IP header. You should see a pattern. Verify that the pattern holds by running the `ssh` command:

```
ssh 192.168.70.5
```

Use the pattern found to write the snort rule. As a hint, you will need rule options found in the Snort manual under section *3.6 Non-Payload Detection Rule Options*.

Verify that your rule detects the port scanning. Verify also that your rule does not generate alerts for benign traffic, such as that generated with ssh and telnet commands above.

Explain in the report all parts of the snort rule, including any options used, and reason why you think the rule will detect only port scanning but not benign traffic.

---

[3] Telnet is used to send commands to a remote terminal. It is similar to ssh, but lacks encryption and other advanced features. Use of telnet is generally discouraged because the traffic between client and server is in plain-text and thus vulnerable to attacks on confidentiality and integrity.

Task 4: Detect DoS attack

Select the `auxiliary/dos/tcp/synflood` module in Metasploit and read its documentation. This is a SYN flooder, which aims to fill up the connection queue at the receiver so that legitimate connections can't be accepted.

Configure the module to attack Server A (192.168.70.5) over interface `enp0s8`. Use Wireshark to study the malicious traffic and identify patterns that can be detected with snort.

HINT: This type of attack is characterized by a high frequency of SYN packets. Read section 3.8 Rule Thresholds in the Snort manual to understand how to use this characteristic to detect the attack.

Verify that your rule(s) detects the port scanning. Verify also that your rule(s) does not generate alerts for benign traffic.

Explain in the report all parts of the snort rule(s), including any options used, and reason why you think the rule(s) will detect only port scanning but not benign traffic.

Task 5: Detect incoming rogue SSH connections

Select the `auxiliary/scanner/ssh/ssh_login` module in Metasploit and read its documentation. This module will cycle through a collection of login names and passwords to test if they can be used to remotely login on a victim machine (Server A, in our case).

Use a text editor and create a file containing the following collection of login names and passwords:

```
Alice 1234
Bob secret
John john
Marty password
Gina anig
```

Save the file under name you like (but keep it simple to type!). In Metasploit assign to the variable USERPASS_FILE the complete path to the file above. Configure also the RHOSTS variable with the address of Server A.

Use Wireshark to study the malicious traffic[4]. Focus on the initial SSH handshake and look at the data exchanged. Use the `ssh` command (as in previous tasks) to create benign SSH traffic to Server A and compare to the malicious traffic. In the handshake, there will be a string that you can use to match the SSH traffic from the rogue client.

[4] You can also examine the tail of the file `/var/log/auth.log`. There you will see evidence of the attack, but sadly that can't be used together with snort.

HINT: Read first 10 sections under *Chapter 3 Writing Snort Rules* in the snort manual to learn how to match content from packets.

Verify that your rule(s) alert only on malicious connections attempt, and that alerts are not generated for benign traffic.

Explain in the report all parts of the snort rule(s), including any options used, and reason why you think the rule(s) will detect only port scanning but not benign traffic.

Deliverables

You are expected to provide an **individual** report in PDF format, where you detail how you solved each task included in the lab guide. Provide screenshots as proof that your commands/changes work as intended. In tasks that required you to make configuration changes (e.g., snort rules), list the modifications you made and explain where you made them (on which VM and in what file). Similarly, mention all settings you modified for a particular Metasploit module.

If you were asked to verify snort rules, explain how the verification was done and provide evidence (e.g., screenshots) that the verification succeeded.

Your report must be written in English. On the first page make sure you include your full name (as it appears in Its Learning), your personal number, and your e-mail address.

In addition to the report, upload also your `snort.conf` file containing all rules created during the lab.