Lab 3

Snort

Contents

3.1 Introduction[1]

Snort is a popular choice for running a network intrusion detection systems or NIDS. It monitors the package data sent and received through a specific network interface. NIDS can catch threats targeting your system vulnerabilities using signature-based detection and protocol analysis technologies. NIDS software, when installed and configured appropriately, can identify the latest attacks, malware infections, compromised systems, and network policy violations.

This lab will be focused on the installation of Snort, its initial configuration and setting rules. For this lab to work you will be asked to register to the Snort community to be able to download the registered user based rules.

[1]Section based on: https://upcloud.com/community/tutorials/install-snort-ubuntu/

3.2 Preparing the VM

For this lab we will use the same VM we used for the IPTables and Certificates labs. You can use directly that one or download a fresh version from: https://softdocencia.fib.upc.edu/software/Ubuntu64-18LTSv1.zip. You can use the username `alumne` and password `sistemes`.

3.3 Installing Snort

3.3.1 Preparing your server

Setting up a basic configuration of Snort on Ubuntu is fairly simple but takes a few steps to complete. You will first need to install all the prerequisite software for installing Snort itself. Install the required libraries with the following command.

```
$ sudo apt install -y gcc libpcre3-dev zlib1g-dev libluajit-5.1-dev
    libpcap-dev openssl libssl-dev libnghttp2-dev libdumbnet-dev bison
     flex libdnet
```

With the prerequisites fulfilled, next up is how to install Snort on your system.

3.3.2 Installing from source

Setting up Snort consists of a series of steps:

- downloading the code,
- configuring it,
- compiling the code,
- installing it to the appropriate directory

Start by making a temporary download folder to your home directory and then changing into it with the command below.

```
$ mkdir ~/snort_src && cd ~/snort_src
```

Snort itself uses something called Data Acquisition library (DAQ) to make abstract calls to packet capture libraries. Download and untar the latest DAQ source package from the Snort website with the `wget` command.

```
$ wget https://www.snort.org/downloads/snort/daq-2.0.7.tar.gz -O - |
    tar -xzf -
$ cd daq-2.0.7
```

Run the configuration script using its default values, then compile the program with make and finally install DAQ.

```
$ ./configure && make && sudo make install
```

With the DAQ installed you can get started with Snort, change back to the download folder.

```
$ cd ~/snort_src
```

Next, download the Snort source code with `wget`. You can find the latest version number on the Snort downloads page. Replace it in the following command if necessary.

```
$ wget https://www.snort.org/downloads/snort/snort-2.9.16.1.tar.gz -O
    - | tar -xzf-
$ cd snort-2.9.16.1
```

Once the download is complete, extract the source and change into the new directory with these commands.

Then configure the installation with `sourcefire` enabled.

```
$ ./configure --enable-sourcefire && make && sudo make install
```

With that done, continue below on how to set up the configuration files.

3.3.3 Configuring Snort to run in NIDS mode

Next, you will need to configure Snort for your system. This includes editing some configuration files, downloading the rules that Snort will follow, and taking Snort for a test run.

Start with updating the shared libraries using the command underneath.

```
$ sudo ldconfig
```

3.3.4 Setting up username and folder structure

To run Snort safely without root access, you should create a new unprivileged user and a new user group for the daemon to run under.

```
$ sudo groupadd snort
$ sudo useradd snort -r -s /sbin/nologin -c SNORT_IDS -g snort
```

Then create the folder structure to house the Snort configuration.

```
$ sudo mkdir -p /etc/snort/rules
$ sudo mkdir /var/log/snort
$ sudo mkdir /usr/local/lib/snort_dynamicrules
```

Set permissions for the new directories accordingly.

```
$ sudo chmod -R 5775 /etc/snort
$ sudo chmod -R 5775 /var/log/snort
$ sudo chmod -R 5775 /usr/local/lib/snort_dynamicrules
$ sudo chown -R snort:snort /etc/snort
$ sudo chown -R snort:snort /var/log/snort
$ sudo chown -R snort:snort /usr/local/lib/snort_dynamicrules
```

Create new files for the white and blacklists as well as local rules.

```
$ sudo touch /etc/snort/rules/white_list.rules
$ sudo touch /etc/snort/rules/black_list.rules
$ sudo touch /etc/snort/rules/local.rules
```

Then copy the configuration files from the download folder.

```
$ sudo cp ~/snort_src/snort-2.9.16.1/etc/*.conf* /etc/snort
$ sudo cp ~/snort_src/snort-2.9.16.1/etc/*.map /etc/snort
```

Next up, you will need to download the detection rules Snort will follow to identify potential threats. Snort provides three tiers of rule sets, community, registered and subscriber rules.

Community rules are freely available though slightly limited. That's why in this lab we will register for free on Snort website, then you get access to the Oink code, which lets you download the registered users rule sets.

Lastly, subscriber rules are just that, available to users with an active subscription to Snort services.

Below you can find instructions for downloading both community rules or registered user rule sets.

3.3.4.1 Obtaining registered user rules

Go to Snort webpage: http://www.snort.org and obtain your `oinkcode`. You can find the code in the Snort user account details.

Replace NNN with the last snortrules number and the oinkcode in the following command with your personal code. Note that there is no space between ? and oinkcode.

```
$ wget https://www.snort.org/rules/snortrules-snapshot-NNN.tar.gz?
    oinkcode=YOUROINKCODE -O - | $ sudo tar -C /etc/snort -xzf -
```

The rule sets for the registered users include an extensive amount of useful preconfigured detection rules. You can enable the additional rules by uncommenting their inclusions at the end of the `snort.conf` file.

3.3.5 Configuring the network and rule sets

With the configuration and rule files in place, edit the `snort.conf` to modify a few parameters. Open the configuration file:

```
$ sudo vi /etc/snort/snort.conf
```

Find these sections shown below in the configuration file and change the parameters to reflect your configuration. You have to change the field `HOME_NET` adding the IP address of your machine replacing the field `MACHINE_IP`. You have to replace the field `MACHINE_IP` with the IP address of your machine.

```
# Setup the network addresses you are protecting
ipvar HOME_NET MACHINE_IP/32
# Set up the external network addresses. Leave as "any" in most
    situations
ipvar EXTERNAL_NET !$HOME_NET
# Path to your rules files (this can be a relative path)
var RULE_PATH /etc/snort/rules
var SO_RULE_PATH /etc/snort/so_rules
var PREPROC_RULE_PATH /etc/snort/preproc_rules
# Set the absolute path appropriately
var WHITE_LIST_PATH /etc/snort/rules
var BLACK_LIST_PATH /etc/snort/rules
```

In the same `snort.conf` file, scroll down to the section 6 and set the output for `unified2` to log under filename of snort.log:

```
# unified2
# Recommended for most installs
output unified2: filename snort.log, limit 128, nostamp,
    mpls_event_types, vlan_event_types
```

Lastly, scroll further down towards the end of the file to find the list of included rule sets. You will need to uncomment the local.rules to allow Snort to load any custom rules. We will need this later on this lab.

```
include $RULE_PATH/local.rules
```

Once you are done with the configuration file, save the changes and exit the editor.

3.3.6 Validating the settings

Your Snort should now be ready to run. Test the configuration using the parameter -T to enable test mode.

```
$ sudo snort -T -c /etc/snort/snort.conf
```

After running the Snort configuration test, mind that it will take some time, you will obtain a large output. You will know the configuration was correct if it ends with:

```
...

Snort successfully validated the configuration!

Snort exiting
```

In case you get an error, the print out should tell you what the problem was and where to fix it. Most likely problems are missing files or folders, which you can usually resolve by either adding any you might have missed in the setup above, or by commenting out unnecessary inclusion lines in the `snort.conf` file. Check the configuration part and try again.

3.4 Example for Adding simple rules

In this section of the lab we will log alerts with a custom detection rule alert on incoming ICMP connections to the `local.rules` file. Open your local rules in a text editor.

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

Then add the following line to the file.

```
alert icmp any any -> $HOME_NET any (msg:"ICMP test"; sid:10000001;
    rev:001;)
```

The rule consists of the following parts:

- action for traffic matching the rule, alert in this case
- traffic protocol, for example, TCP, UDP or ICMP like here
- the source address and port, simply marked as any to include all addresses and ports
- the destination address and port, `$HOME_NET` (as declared in the configuration) and `any` respectively
- log messages set to `msg :" ICMP test "`
- unique rule identifier `(sid)` which for local rules needs to be 10000001 or higher
- rule version number, set to : 001

Save the `local.rules` and exit the editor.

Start Snort with `-A` console options to print the alerts to `stdout`. You will need to select the correct network interface with the public IP address of your server, for example, `eth0`.

```
$ sudo snort -A console -i <net_iface> -u snort -g snort -c /etc/
    snort/snort.conf
```

You can obtain your network interface `<net_iface>` using the command:

```
$ ip addr
```

The output will list all of your currently configured network interfaces. Find the one with the same public IP address as shown in the Network settings.

With Snort up and running, ping your cloud server from any other computer. You should see a notice for each ICMP call in the terminal running Snort.

```
07/12-11:20:33.501624  [**] [1:10000001:1] ICMP test [**] [Priority:
    0] {ICMP} 83.136.252.119 -> 80.69.173.202
After the alerts show up you can stop Snort with ctrl+C.
```

Snort records the alerts to a log under `/var/log/snort/snort.log.<timestamp>`, where the time stamp is the point in time when Snort was started marked in Unix time. You can read the logs with the command below. Since you have only run Snort once, there is only one log.

```
$ sudo snort -r /var/log/snort/snort.log.<timestamp>
```

The log shows a warning for each ICMP call with source and destination IPs, time and date, plus some additional info as shown in the example below.

```
WARNING: No preprocessors configured for policy 0.
07/12-11:20:33.501624 83.136.252.118 -> 80.69.173.202
ICMP TTL:63 TOS:0x0 ID:20187 IpLen:20 DgmLen:84 DF
Type:8 Code:0 ID:13891 Seq:1 ECHO
Running Snort in the background
```

3.5 Add more complex rules

The rest of the lab will set a few requirements and the goal is to setup the rules that comply with them.

3.5.1 Detecting DoS

The first exercise is to provide the necessary rules to detect remote Denial of Service (DoS) attacks, in particular we will focus on SYN flooding.

A SYN flooding attack is based on sending a burst of SYN packets to a particular service, forcing it to spawn many threads and potentially rendering the service unavailable due to limits on the possible concurrent number of connections.

In this lab we will focus on the detection of the attacks, not covering Snort mitigation capabilities.

3.5.1.1 Setting up the rules

First step to detect attacks we have to set a Snort rule that allows the detection of the attack.

You have to complete the rule on your own, here you have some hints on how you can achieve this:

- The rule needs to detect SYN packets
- You have to track the connections by Source
- You have to count the number of connections during a short time interval to raise the warning and the log input
- Remember that the `sid` needs to be unique

3.5.1.2 Simulating a DoS attack

To simulate a DoS attack you have to install the `hping3` tool, you can achieve this by running:

```
$ sudo apt-get install hping3
```

Once installed you can check the man page:

```
$ man hping3
```

Now you can generate an attack by creating 1000 concurrent connections to the localhost.

Validate that the attack is detected by Snort.

3.5.2 Detecting DDoS

Similarly to the previous section, in this exercise it is expected that you setup a rule that is able to detect a Distributed Denial of Service (DDoS). It is easy to identify a DDoS, the main difference is that instead of being triggered from a single IP address it is performed coordinately from various sources simultaneously.

For the exercise you have to update the rule on the previous section to cover this new use case.

Hint: try filtering from destination rather than source to be able to detect the attack easily.

You can test the new rule by restarting Snort and using `hping` as in the previous exercise.

3.5.3 Domain detection

To complete this lab, the last step will be to create a set of rules that detect accesses to different web sites:

- `facebook.com`
- `youtube.com`
- `twitter.com`

You have to create the rules that insert on Snort log the message: *`website` access detected*. Where `website` is one of `facebook.com`, `youtube.com`, `twitter.com`.

3.6 Optional configuration

If you want to run Snort as a background process, you will need to create a service on `systemd`. We will create a new file in a text editor for example with the next command.

```
$ sudo vi /lib/systemd/system/snort.service
```

Enter the following to the file, save and exit the editor.

```
[Unit]
Description=Snort NIDS Daemon
After=syslog.target network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/snort -q -u snort -g snort -c /etc/snort/
    snort.conf -i <iface_name>

[Install]
WantedBy=multi-user.target
```

With the service defined, reload the `systemctl` daemon. To make it aware of the changes.

```
$ sudo systemctl daemon-reload
```

Snort can then be now run with the configuration you set up using the command below.

```
$ sudo systemctl start snort
```

The startup script also includes other usual `systemctl` commands: stop, restart, and status. For example, you can check the status of the service with the following command.

```
$ sudo systemctl status snort
```