

# FINAL REPORT

Subjects: Quản trị mạng và hệ thống

Topic : Snort

GVHD: Trần Thị Dung

## 1.GENERAL INFORMATION:

Class : NT132.N12.ATCL

STT	Họ và tên	MSSV	Email
1	Trương Văn Rõng	20521831	20521831@gm.uit.edu.vn
2	Lương Mạnh Tiến	20522008	20522008@gm.uit.edu.vn
3	Phan Hoàng Nam	20521635	20521635@gm.uit.edu.vn

## 2.CONTENTS:

STT	Công việc	Kết quả tự đánh giá
1	I.Introduction	100%
2	1.1 Overview	100%
3	1.2 Component	100%
4	1.3 Operation	100%
5	II. Implementation	100%
6	2.1 Topology	100%
7	2.2 Installation	100%
8	2.3 Configuration	100%
9	III. Result	100%
10	3.1 Basic model	100%
11	3.2 Advance model	100%
12	Task	100%
13	Self - Assign	100%

# Table Of Contents

I. Introduction .....	2
1.1 Overview .....	2
1.2 Component .....	2
1.3 Operation .....	3
II. Implementation .....	4
2.1 Topology .....	4
2.2 Installation .....	5
2.3 Configuration .....	9
III.Result .....	10
❖ Basic result: .....	10
❖ Advance result: .....	13
Task .....	19
Self Assign .....	19
Answer .....	19

# REPORT IN DETAIL

## I. Introduction

### 1.1 Overview

Snort is a free open source Network Intrusion Detection System(NIDS) and Intrusion Prevention System (IPS) which is capable of performing real-time traffic analysis and packet logging on IP networks. It helps define malicious network activity and uses those rules to find packets that match against them and generates alerts for users, also prevent the intrusion.

Snort can be deployed inline to stop these packets, as well. Snort has three primary uses: As a packet sniffer like tcpdump, as a packet logger — which is useful for network traffic debugging, or it can be used as a full-blown network intrusion prevention system. Snort can be downloaded and configured for personal and business use alike.

### 1.2 Component

Snort is comprised of two major components: 5 modules and Rulesets.

5 modules are:

- Sniffer Module.
- Pre-Processor Module.
- Detect Engine Module.
- Alert and Log Module.
- Import/Export data Module

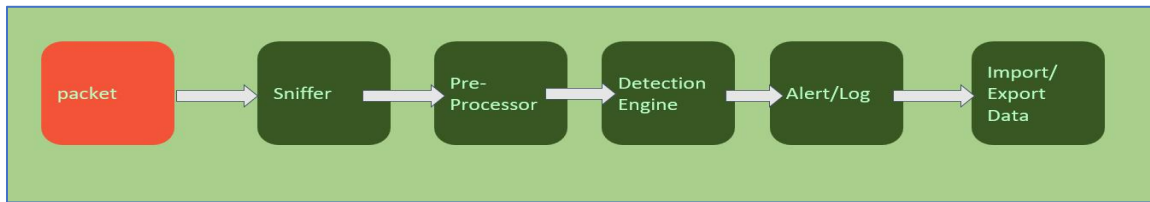
Rulesets:

Rule is a set of description languages, it works with the detection engine to detect the intrusion.

Rules can be written in `/etc/snort/rules/local.rules`

### 1.3 Operation

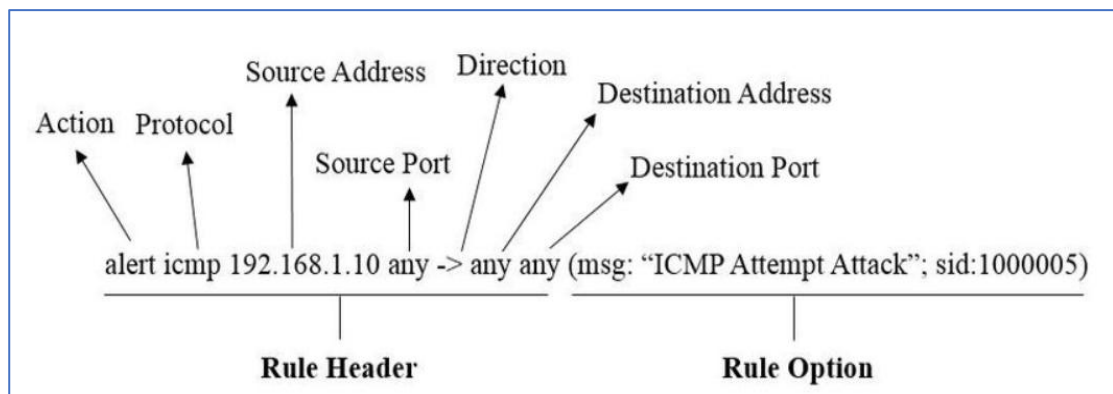
We can understand how Snort apply 5 modules by this image



Packets which were sniffed by Snort will go into Sniffer module, then go into pre-processor to decode or format things. After that, it will go through the detection module, if there is any intrusion, it will pass to the alert/log module then eventually import or export the data.

Snort IPS uses rules that aids in the definition of malicious network activity and employs those rules to find packets that match against them, generating alerts for users

Each rule has a structure like this:



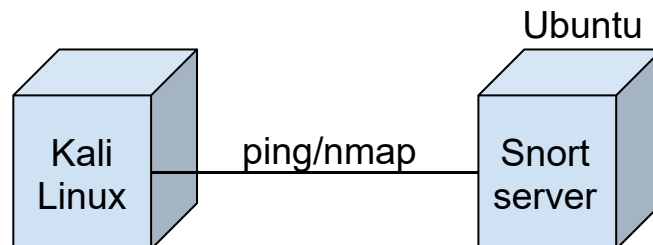
**Snort provides severals mode to operate:**

- Sniffer mode.
- Logging mode.
- Network intrusion detection system mode.
- IPS Inline Mode

## II. Implementation

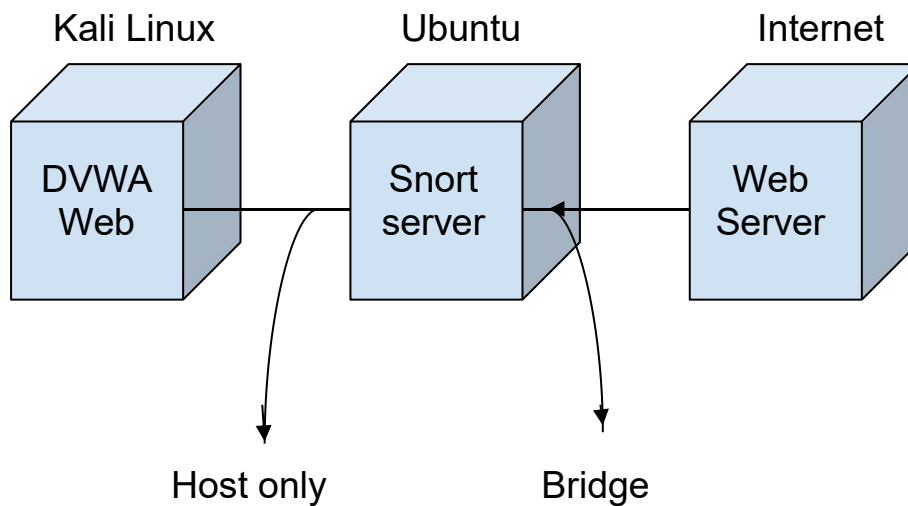
### 2.1 Topology

In basic model:



- ❖ I configured the NAT network in VMware's Network Adapter in both Kali and Ubuntu machines.

In advanced model:



- ❖ I configured the Bridge network in VMware's Network Adapter between the internet and Ubuntu machine
- ❖ In VMware's Network Adapter, I set up a Host-only network between the Ubuntu system and Kali Linux.

	IP	Network
Ubuntu	192.168.73.128	Host only/ Bridge
Kali	192.168.73.129	Host only
Host	192.168.73.1	

## 2.2 Installation

- **Server:** Lubuntu (install snort)
- **Attacker:** Kali (use malicious network attacks)

### ❖ In Ubuntu server

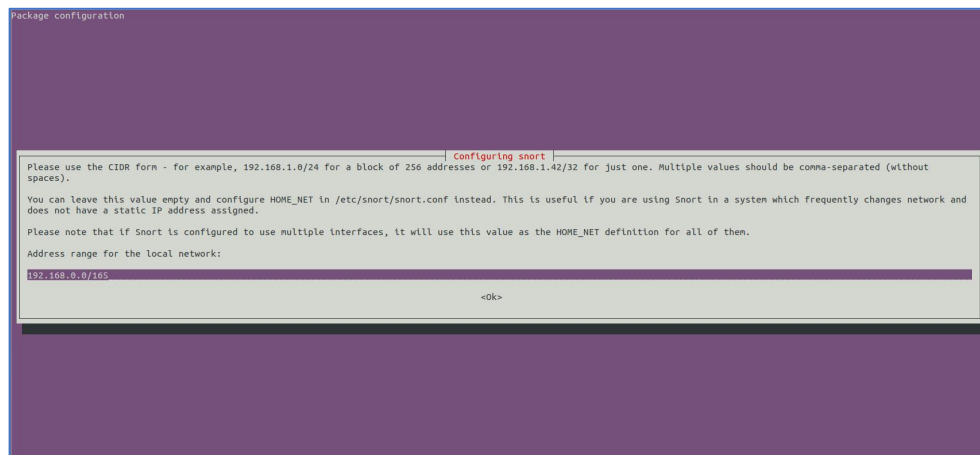
**Step 1:** Before installing snort, make sure you have dev packages of libpcap and libpcrc and have the latest apt install packet.

Use this command:

```
# apt-cache policy libpcap0.8-dev
libpcap0.8-dev:
  Installed: 1.0.0-2ubuntu1
  Candidate: 1.0.0-2ubuntu1

# apt-cache policy libpcrc3-dev
libpcrc3-dev:
  Installed: 7.8-3
  Candidate: 7.8-3
```

Then find the appropriate package for the operating system and install snort.



Select the address range for the local network and click Ok.

```
nam@nam-virtual-machine:~$ snort
Running in packet dump mode

--== Initializing Snort ==--
Initializing Output Plugins!
pcap DAQ configured to passive.
Acquiring network traffic from "ens33".
ERROR: Can't start DAQ (-1) - socket: Operation not permitted!
Fatal Error, Quitting..
```

After successful installation. Check Snort version

```
# snort --version
```

```
nam@nam-virtual-machine:~$ snort --version

  ,,-_
 o" )~
  ' '

-*> Snort! <*-
Version 2.9.15.1 GRE (Build 15125)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014-2019 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.10.1 (with TPACKET_V3)
Using PCRE version: 8.39 2016-06-14
Using ZLIB version: 1.2.11
```

❖ **Next step, Download and add snort rules**

- To make the snort tool work, we need add the rules of snort.
- Can be downloaded directly on the snort site (supported by the community) using wget and saved as the community.tar.gz file

```
wget https://www.snort.org/rules/community -O ~/community.tar.gz
```

Extract the file with the tar . command

```
nam@nam-virtual-machine:~$ ls community.tar.gz
community.tar.gz
nam@nam-virtual-machine:~$ tar xvzf community.tar.gz
community-rules/
community-rules/community.rules
community-rules/VRT-License.txt
community-rules/LICENSE
community-rules/AUTHORS
community-rules/snort.conf
community-rules/sid-msg.map
nam@nam-virtual-machine:~$
```

As a result, we can have the community-rules . directory:

Bash ▾

```
sudo cp ~/community-rules/* /etc/snort/rules
```

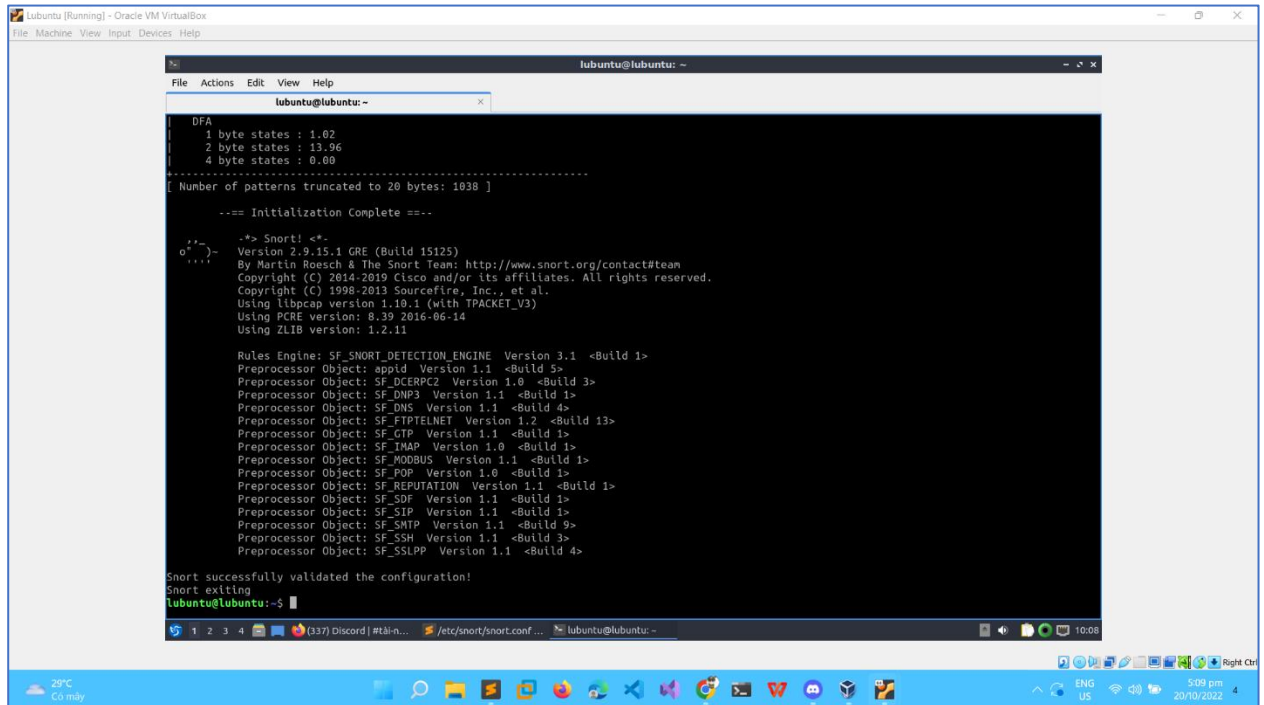
```
nam@nam-virtual-machine:~$ ls /etc/snort/rules/
attack-responses.rules  community-mail-client.rules  community-web-iis.rules  info.rules  porn.rules  web-attacks.rules
AUTHORS                 community-misc.rules         community-web-misc.rules  LICENSE     rpc.rules  web-cgi.rules
backdoor.rules          community-nntp.rules         community-web-php.rules  local.rules  rservices.rules  web-client.rules
bad-traffic.rules       community-oracle.rules      ddos.rules              misc.rules  scan.rules  web-coldfusion.rules
chat.rules              community-policy.rules      deleted.rules            multimedia.rules  sid-msg.map  web-frontpage.rules
community-bot.rules     community-rules             dns.rules               mysql.rules  snmp.rules  web-iis.rules
community-deleted.rules community-sip.rules          dos.rules               netbios.rules  smtp.rules  web-misc.rules
community-dos.rules     community-sql.rules         exploit.rules            nntp.rules   snort.conf  web-php.rules
community-exploit.rules community-sql-injection.rules  finger.rules            oracle.rules  sql.rules   x11.rules
community-ftp.rules     community-virus.rules       ftp.rules               other-ids.rules  telnet.rules
community-game.rules    community-web-attacks.rules  icmp-info.rules         p2p.rules    tftp.rules
community-icmp.rules    community-web-cgi.rules      icmp.rules              policy.rules  virus.rules
community-imap.rules    community-web-client.rules   imap.rules              pop2.rules   VRT-License.txt
community-inappropriate.rules
nam@nam-virtual-machine:~$
```



## Test Snort:

Bash ▾

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



```
lubuntu@lubuntu: ~  
DFA  
1 byte states : 1.02  
2 byte states : 13.96  
4 byte states : 0.00  
-----  
[ Number of patterns truncated to 20 bytes: 1038 ]  
-----  
...== Initialization Complete ==...  
o'-'> Snort! <-'  
o'-'> Version 2.9.15.1 GRE (Build 15125)  
o'-'> By Martin Roesch & The Snort Team: http://www.snort.org/contact#team  
o'-'> Copyright (C) 2014-2019 Cisco and/or its affiliates. All rights reserved.  
o'-'> Copyright (C) 1998-2013 Sourcefire, Inc., et al.  
o'-'> Using libpcap version 1.10.1 (with TPACKET_V3)  
o'-'> Using PCRE version: 8.39 2016-06-14  
o'-'> Using ZLIB version: 1.2.11  
Rules Engine: SF_SNORT_DETECTION_ENGINE Version 3.1 <Build 1>  
Preprocessor Object: appid Version 1.1 <Build 5>  
Preprocessor Object: SF_DCEPC2 Version 1.0 <Build 3>  
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>  
Preprocessor Object: SF_DNS Version 1.1 <Build 4>  
Preprocessor Object: SF_FIPIELNET Version 1.2 <Build 13>  
Preprocessor Object: SF_GTP Version 1.1 <Build 1>  
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>  
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>  
Preprocessor Object: SF_POP Version 1.0 <Build 1>  
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>  
Preprocessor Object: SF_SDF Version 1.1 <Build 1>  
Preprocessor Object: SF_SIP Version 1.1 <Build 1>  
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>  
Preprocessor Object: SF_SSH Version 1.1 <Build 3>  
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>  
Snort successfully validated the configuration!  
Snort exiting  
lubuntu@lubuntu:~$
```

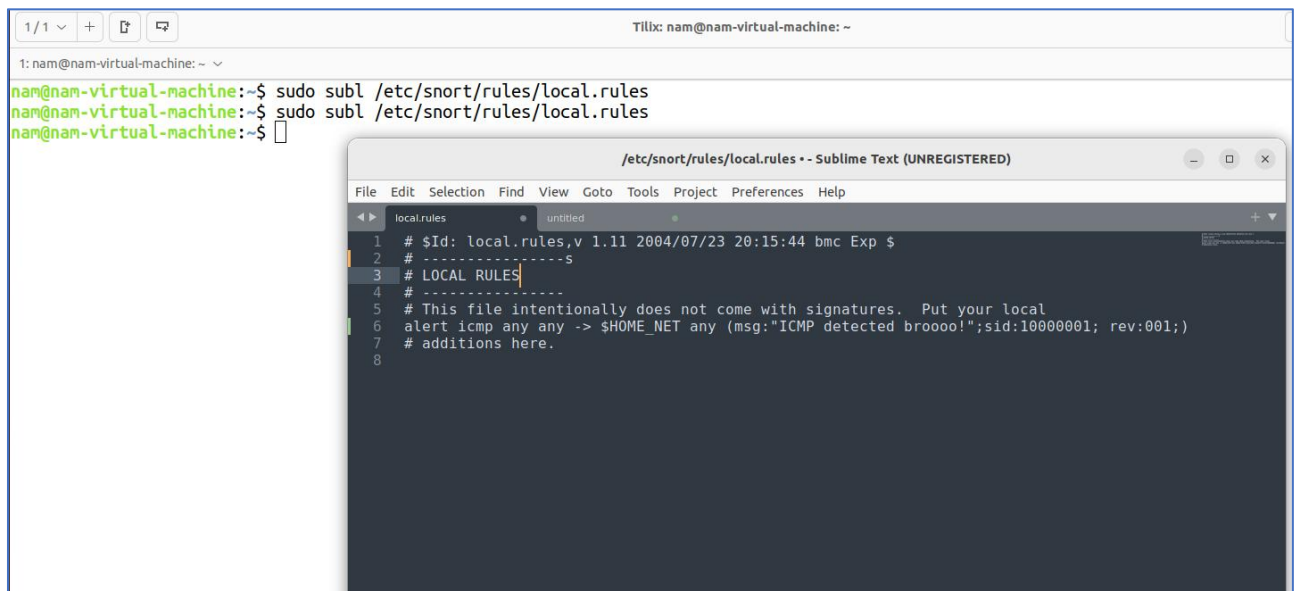
## 2.3 Configuration

Add this rule to file locals.local

Bash ▾

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

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



The screenshot shows a terminal window with the following commands and output:

```
1: nam@nam-virtual-machine: ~ ▾
nam@nam-virtual-machine:~$ sudo subl /etc/snort/rules/local.rules
nam@nam-virtual-machine:~$ sudo subl /etc/snort/rules/local.rules
nam@nam-virtual-machine:~$
```

The Sublime Text editor window, titled "/etc/snort/rules/local.rules - Sublime Text (UNREGISTERED)", displays the content of the file:

```
1 # $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
2 # -----s
3 # LOCAL RULES
4 # -----
5 # This file intentionally does not come with signatures. Put your local
6 alert icmp any any -> $HOME_NET any (msg:"ICMP detected broooo!";sid:10000001; rev:001;)
7 # additions here.
8
```

We try to ping and see result

```

lubuntu@lubuntu:~$ sudo subl /etc/snort/rules/local.rules
lubuntu@lubuntu:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::7e08:bfd7:3342:8e0f prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:c3:00:2c txqueuelen 1000 (Ethernet)
    RX packets 91146 bytes 120522954 (120.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 22640 bytes 3186838 (3.1 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1323 bytes 145775 (145.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1323 bytes 145775 (145.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lubuntu@lubuntu:~$ ping 10.0.2.5
PING 10.0.2.5 (10.0.2.5) 56(84) bytes of data:
64 bytes from 10.0.2.5: icmp_seq=1 ttl=64 time=1.00 ms
64 bytes from 10.0.2.5: icmp_seq=2 ttl=64 time=0.715 ms
64 bytes from 10.0.2.5: icmp_seq=3 ttl=64 time=0.705 ms
64 bytes from 10.0.2.5: icmp_seq=4 ttl=64 time=0.794 ms
^C
--- 10.0.2.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3331ms
rtt min/avg/max/mdev = 0.705/0.803/1.000/0.118 ms
lubuntu@lubuntu:~$
  
```

### III.Result

#### ❖ Basic result:

Kali VM:

- In Kali machine I use the ping command and ping to Ubuntu machine
- The ping command use ICMP protocol

```

(yugeiv3@yugeiv3)-[~]
$ ping 192.168.73.128
PING 192.168.73.128 (192.168.73.128) 56(84) bytes of data:
64 bytes from 192.168.73.128: icmp_seq=1 ttl=64 time=0.386 ms
64 bytes from 192.168.73.128: icmp_seq=2 ttl=64 time=0.396 ms
64 bytes from 192.168.73.128: icmp_seq=3 ttl=64 time=0.367 ms
64 bytes from 192.168.73.128: icmp_seq=4 ttl=64 time=0.469 ms
^C
— 192.168.73.128 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3066ms
rtt min/avg/max/mdev = 0.367/0.404/0.469/0.038 ms
  
```

Ubuntu VM:

- I run this command to detect the ping command from kali machine

```
ubuntu@ubuntu:~$ snort -V

o''~
''''

-*> Snort! <*-
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 1.9.1 (with TPACKET_V3)
Using PCRE version: 8.39 2016-06-14
Using ZLIB version: 1.2.11

ubuntu@ubuntu:~$ sudo snort -d -l /var/log/snort/ -h 192.168.1.0/24 -A console -c /etc/snort/snort.conf
```

- When it receive the ping from kali machine, the snort can detect the malicious packet and then display in the screen (includes source IP, destination IP, protocol and the network attack command)

```
11/24-10:44:22.839070  [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.73.129 -> 192.168.73.128
11/24-10:44:22.839070  [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.73.129 -> 192.168.73.128
11/24-10:44:22.839088  [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.73.128 -> 192.168.73.129
11/24-10:44:23.863099  [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.73.129 -> 192.168.73.128
11/24-10:44:23.863099  [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.73.129 -> 192.168.73.128
11/24-10:44:23.863132  [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.73.128 -> 192.168.73.129
11/24-10:44:36.375381  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification: Detection of a Network Scan] [Priority: 3] {UDP} 192.168.73.1:58204 -> 239.255.255.250:1900
11/24-10:44:37.380878  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification: Detection of a Network Scan] [Priority: 3] {UDP} 192.168.73.1:58204 -> 239.255.255.250:1900
11/24-10:44:38.388182  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification: Detection of a Network Scan] [Priority: 3] {UDP} 192.168.73.1:58204 -> 239.255.255.250:1900
11/24-10:44:39.397446  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification: Detection of a Network Scan] [Priority: 3] {UDP} 192.168.73.1:58204 -> 239.255.255.250:1900
```

- After that I write the rule by myself to reject the ping command from kali machine by blocking that IP

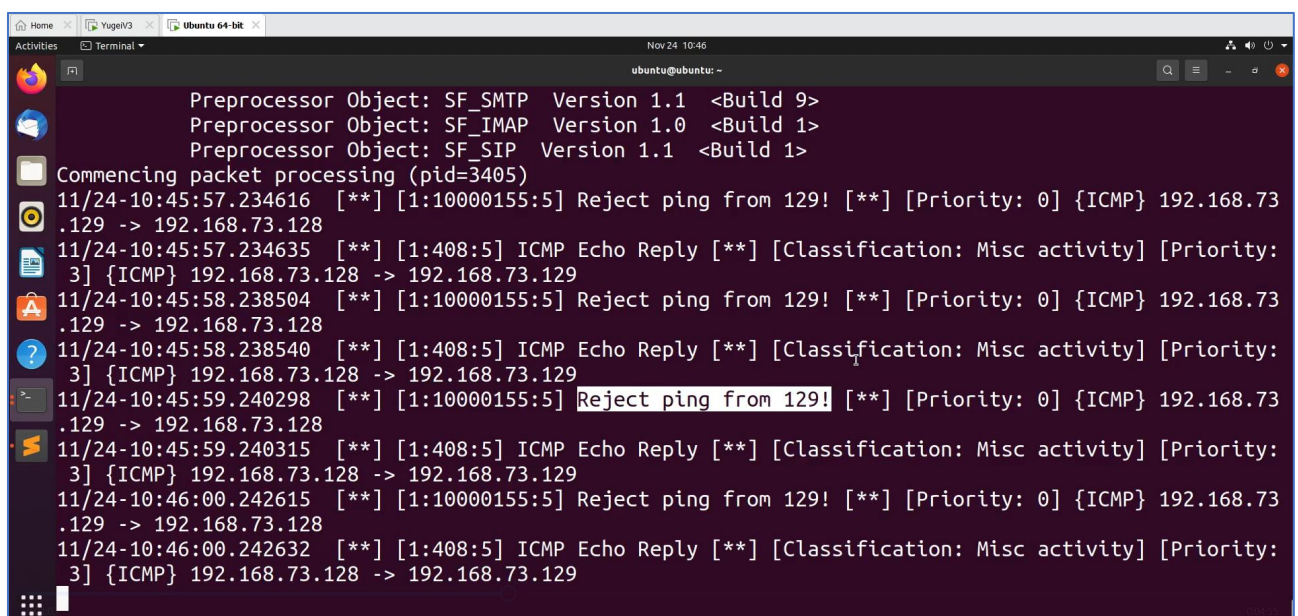
```
3 # LOCAL RULES
4 # -----
5 # This file intentionally does not come with signatures. Put your local
6 # additions here.
7 # alert icmp any any -> $HOME_NET any (msg:"ICMP detected broooo !";sid:10000001; rev:001;)
8 # alert udp 10.0.22.115 any -> 192.168.80.128 any (msg:"ICMP detected broooo !";sid:10000001; rev:001;)
9 # alert udp any any -> any any (msg:"UDP detected broooo !";sid:10000002; rev:002;)
10
11 #REJECT
12 #ping
13 reject icmp 192.168.73.129 any -> 192.168.73.128 any (msg:"Reject ping from 129!"; sid:10000155; rev:005;)
14
15 #External
16 #reject tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "192.168.73.1 has been blocked !"; sid:100000110; )
17
```



- Then I save the rule and run snort again
- Now in Kali machine can not ping to the Ubuntu machine

```
(yugeiv3@yugeiv3)-[~]
$ ping 192.168.73.128
PING 192.168.73.128 (192.168.73.128) 56(84) bytes of data.
64 bytes from 192.168.73.128: icmp_seq=1 ttl=64 time=0.371 ms
From 192.168.73.128 icmp_seq=1 Destination Port Unreachable
64 bytes from 192.168.73.128: icmp_seq=2 ttl=64 time=0.404 ms
From 192.168.73.128 icmp_seq=2 Destination Port Unreachable
64 bytes from 192.168.73.128: icmp_seq=3 ttl=64 time=0.377 ms
From 192.168.73.128 icmp_seq=3 Destination Port Unreachable
64 bytes from 192.168.73.128: icmp_seq=4 ttl=64 time=0.319 ms
From 192.168.73.128 icmp_seq=4 Destination Port Unreachable
^C
— 192.168.73.128 ping statistics —
4 packets transmitted, 4 received, +4 errors, 0% packet loss, time 3008ms
rtt min/avg/max/mdev = 0.319/0.367/0.404/0.030 ms
```

- In Ubuntu machine, it rejected the ping command from Kali machine



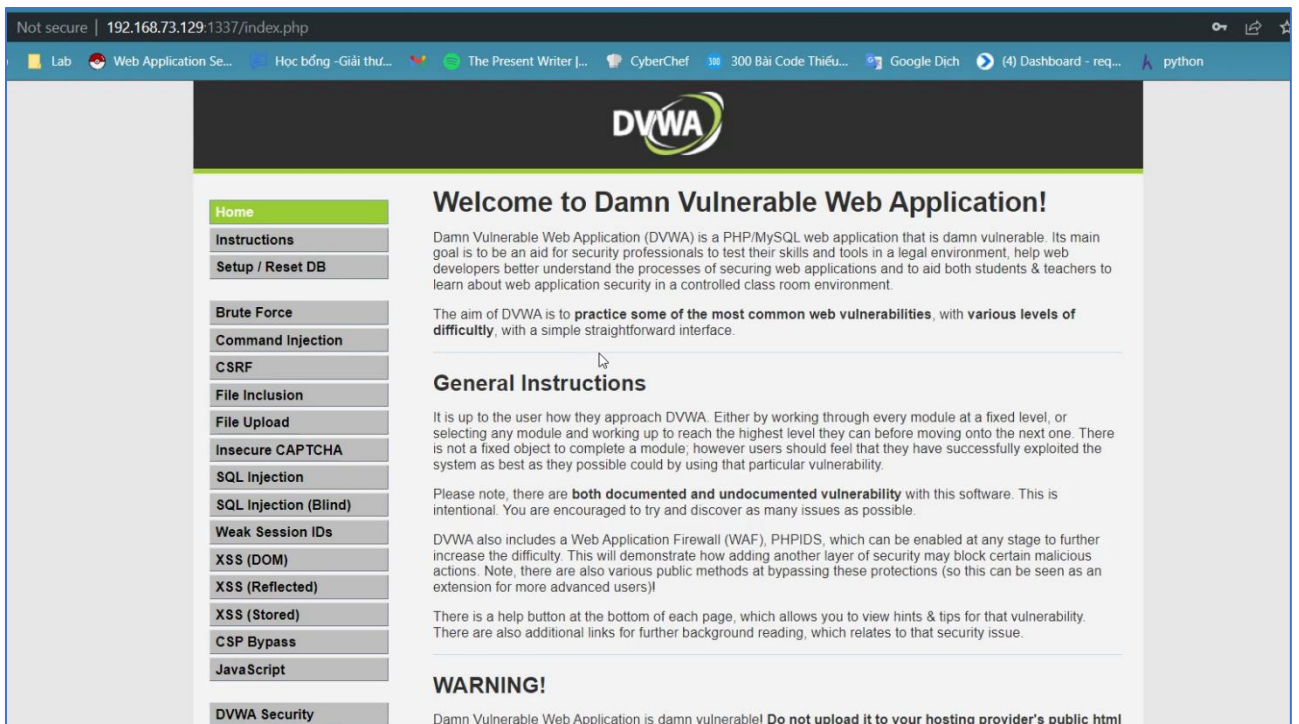
```
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Commencing packet processing (pid=3405)
11/24-10:45:57.234616 [**] [1:10000155:5] Reject ping from 129! [**] [Priority: 0] {ICMP} 192.168.73.129 -> 192.168.73.128
11/24-10:45:57.234635 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.73.128 -> 192.168.73.129
11/24-10:45:58.238504 [**] [1:10000155:5] Reject ping from 129! [**] [Priority: 0] {ICMP} 192.168.73.129 -> 192.168.73.128
11/24-10:45:58.238540 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.73.128 -> 192.168.73.129
11/24-10:45:59.240298 [**] [1:10000155:5] Reject ping from 129! [**] [Priority: 0] {ICMP} 192.168.73.129 -> 192.168.73.128
11/24-10:45:59.240315 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.73.128 -> 192.168.73.129
11/24-10:46:00.242615 [**] [1:10000155:5] Reject ping from 129! [**] [Priority: 0] {ICMP} 192.168.73.129 -> 192.168.73.128
11/24-10:46:00.242632 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.73.128 -> 192.168.73.129
```

Thus, we have successfully detected and reject the attack using the ping command

## ❖ Advance result:

- In Kali machine I hosts the DVWA website by using docker

```
(yugeiv3@yugeiv3)-[~]
$ sudo docker run -p 1337:80 ab0d83586b6e
[+] Starting mysql...
Starting MariaDB database server: mysqld.
[+] Starting apache
Starting Apache httpd web server: apache2AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using
172.17.0.2. Set the 'ServerName' directive globally to suppress this message
.
=> /var/log/apache2/access.log <=
=> /var/log/apache2/error.log <=
[Thu Nov 24 18:46:47.226475 2022] [mpm_prefork:notice] [pid 281] AH00163: Apache/2.4.25 (Debian) configured -- resuming normal operation
[Thu Nov 24 18:46:47.226678 2022] [core:notice] [pid 281] AH00094: Command line: '/usr/sbin/apache2'
=> /var/log/apache2/other_vhosts_access.log <=
```



- I write the rule by myself to detect the TCP protocol:

```
File Edit Selection Find View Goto Tools Project Preferences Help
local.rules
15 #External
16 #reject tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "192.168.73.1 has been blocked !"; sid:100000110; )
17
18
19 # TCP
20 alert tcp 192.168.73.1 any -> 192.168.73.129 any (content:"HTTP"; msg:"HTTP detected broooo !";
  sid:10000100; rev:005;)
21 # UDP
22 alert udp 192.168.73.1 any -> 192.168.73.129 any (content:"HTTP"; msg:"HTTP detected broooo !";
  sid:10000101; rev:005;)
23
```

- Then I write some rules to detect the common vulnerabilities
  - Such as SQLI, OS Command Injection, XSS
- ✓ I attack OS Command Injection in the website

## Vulnerability: Command Injection

### Ping a device

Enter an IP address:

help  
index.php  
source

### More Information

- <http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
- <http://www.ss64.com/bash/>
- <http://www.ss64.com/nt/>
- [https://www.owasp.org/index.php/Command\\_Injection](https://www.owasp.org/index.php/Command_Injection)

## Vulnerability: Command Injection

### Ping a device

Enter an IP address:

**Vulnerability: Command Injection**

Ping a device

Enter an IP address:

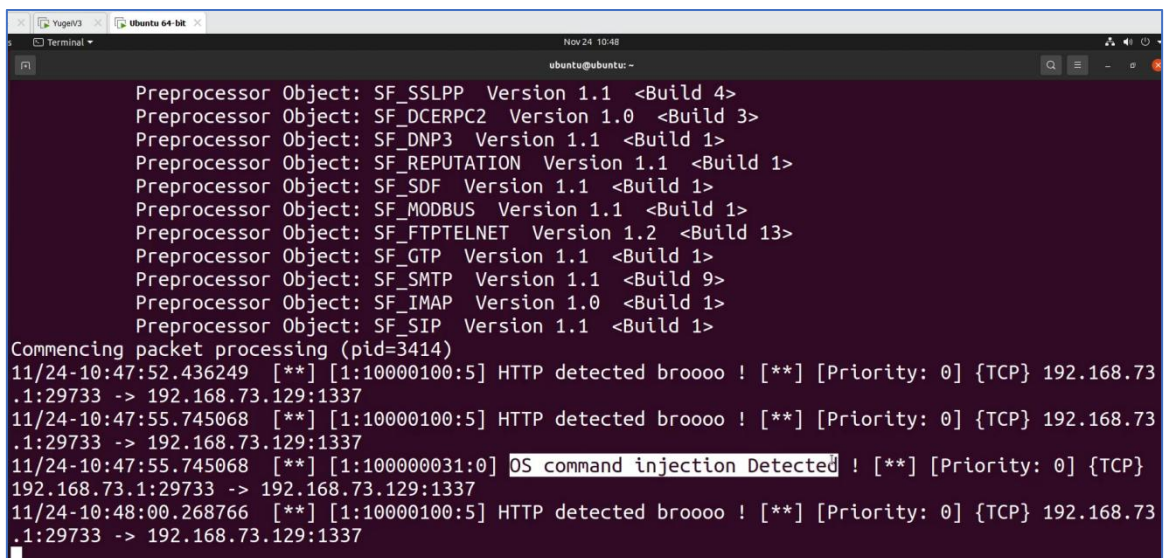


- Snort can detect the Os Command Injection by the rules which I wrote

```

56 # OS command injection
57 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "OS command injection Detected !"; content: "cats" ;
sid:100000030; )
58 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "OS command injection Detected !"; content: "ls" ;
sid:100000031; )
59 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "OS command injection Detected !"; content: "subl" ;
sid:100000032; )
60 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "OS command injection Detected !"; content: "ps" ;
sid:100000033; )
61 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "OS command injection Detected !"; content: "ping" ;
sid:100000034; )
62 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "OS command injection Detected !"; content: "echo" ;
sid:100000035; )
63 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "OS command injection Detected !"; content: "\\x00"

```




```

Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>
Preprocessor Object: SF_DNP3 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_MODBUS Version 1.1 <Build 1>
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Commencing packet processing (pid=3414)
11/24-10:47:52.436249 192.168.73.1:29733 -> 192.168.73.129:1337 [**] [1:10000100:5] HTTP detected broooo ! [**] [Priority: 0] {TCP} 192.168.73.1:29733 -> 192.168.73.129:1337
11/24-10:47:55.745068 192.168.73.1:29733 -> 192.168.73.129:1337 [**] [1:10000100:5] HTTP detected broooo ! [**] [Priority: 0] {TCP} 192.168.73.1:29733 -> 192.168.73.129:1337
11/24-10:47:55.745068 192.168.73.1:29733 -> 192.168.73.129:1337 [**] [1:100000031:0] OS command injection Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29733 -> 192.168.73.129:1337
11/24-10:48:00.268766 192.168.73.1:29733 -> 192.168.73.129:1337 [**] [1:10000100:5] HTTP detected broooo ! [**] [Priority: 0] {TCP} 192.168.73.1:29733 -> 192.168.73.129:1337

```

- ✓ I attack SQLi in the website



**DVWA**

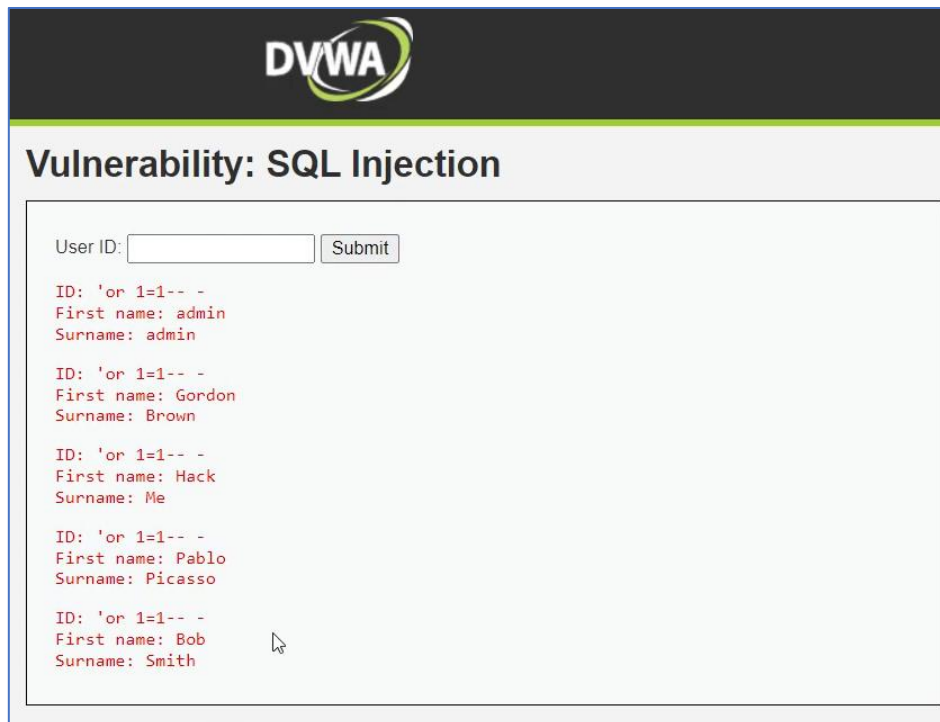
## Vulnerability: SQL Injection

User ID:

### More Information

- <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
- [https://en.wikipedia.org/wiki/SQL\\_injection](https://en.wikipedia.org/wiki/SQL_injection)
- <http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>
- <http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>
- [https://www.owasp.org/index.php/SQL\\_injection](https://www.owasp.org/index.php/SQL_injection)
- <http://bobby-tables.com/>





The image shows the DVWA (Damn Vulnerable Web Application) interface for the 'Vulnerability: SQL Injection' section. At the top, there is a DVWA logo. Below it, the title 'Vulnerability: SQL Injection' is displayed. The main area contains a form with a 'User ID:' label and a text input field, followed by a 'Submit' button. Below the form, there are five rows of output, each starting with 'ID: 'or 1=1-- -'. The first row shows 'First name: admin' and 'Surname: admin'. The second row shows 'First name: Gordon' and 'Surname: Brown'. The third row shows 'First name: Hack' and 'Surname: Me'. The fourth row shows 'First name: Pablo' and 'Surname: Picasso'. The fifth row shows 'First name: Bob' and 'Surname: Smith'.

➤ Snort can detect the SQLi by the rules which I wrote

```

24 # SQLI
25 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "SQL Injection Detected !"; content: "- - - ";
26   sid:100000001; )
27 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "SQL Injection Detected !"; content: "%2D" ;
28   sid:100000002; )
29 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "SQL Injection Detected !"; content: "%27" ;
30   sid:100000003; )
31 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "SQL Injection Detected !"; content: "%22" ;
32   sid:100000004; )
33
34

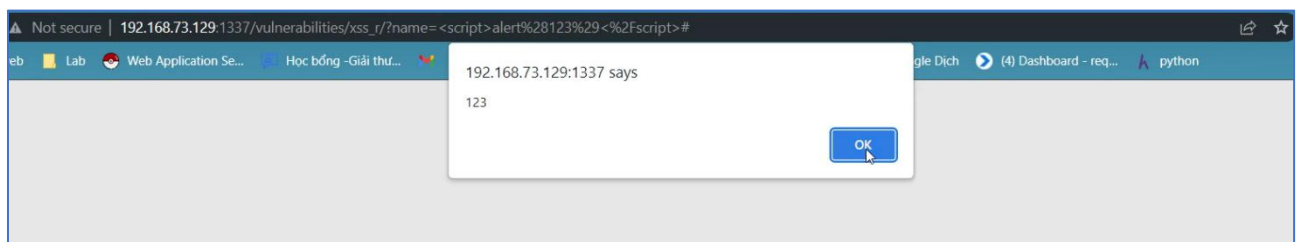
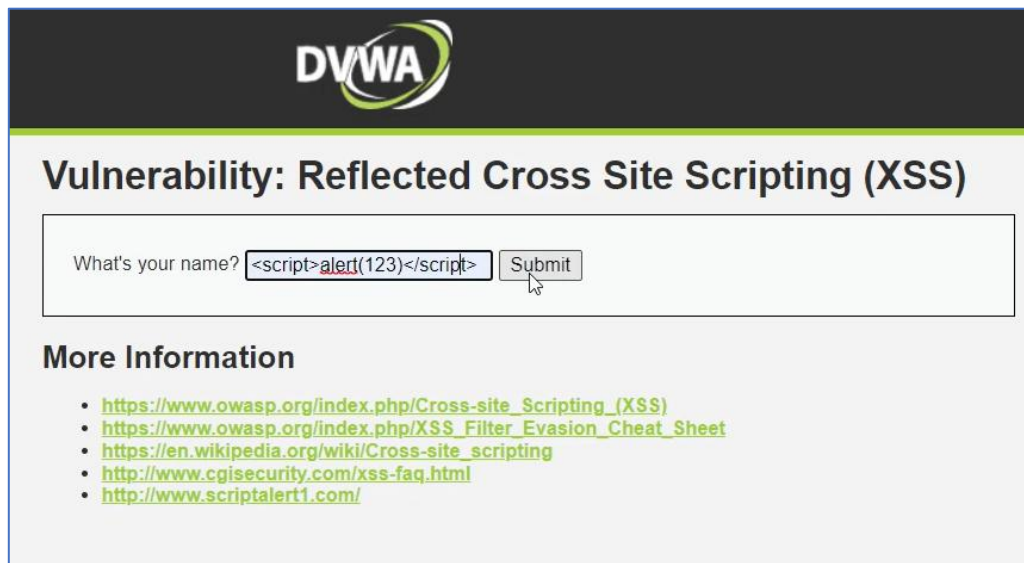
```

```

11/24-10:47:55.745068  [**] [1:100000031:0] OS command injection Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29733 -> 192.168.73.129:1337
11/24-10:48:00.268766  [**] [1:10000100:5] HTTP detected broooo ! [**] [Priority: 0] {TCP} 192.168.73.1:29733 -> 192.168.73.129:1337
11/24-10:48:36.388473  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification: Detection of a Network Scan] [Priority: 3] {UDP} 192.168.73.1:56598 -> 239.255.255.250:1900
11/24-10:48:37.394954  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification: Detection of a Network Scan] [Priority: 3] {UDP} 192.168.73.1:56598 -> 239.255.255.250:1900
11/24-10:48:38.403054  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification: Detection of a Network Scan] [Priority: 3] {UDP} 192.168.73.1:56598 -> 239.255.255.250:1900
11/24-10:48:39.409837  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification: Detection of a Network Scan] [Priority: 3] {UDP} 192.168.73.1:56598 -> 239.255.255.250:1900
11/24-10:48:58.138445  [**] [1:10000100:5] HTTP detected broooo ! [**] [Priority: 0] {TCP} 192.168.73.1:29740 -> 192.168.73.129:1337
11/24-10:49:02.948504  [**] [1:10000100:5] HTTP detected broooo ! [**] [Priority: 0] {TCP} 192.168.73.1:29741 -> 192.168.73.129:1337
11/24-10:49:02.948504  [**] [1:100000003:0] SQL Injection Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29741 -> 192.168.73.129:1337
11/24-10:49:02.948504  [**] [1:100000001:0] SQL Injection Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29741 -> 192.168.73.129:1337

```

- ✓ Finally, I attack the website by XSS vulnerability



- ✓ Snort can detect XSS vulnerability

```

31 # Stored Cross Site Scripting (XSS)
32 # XSS Reflect
33 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "Reflected XSS Detected !"; content: "alert(" ;
  sid:100000010; )
34 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "Reflected XSS Detected !"; content: "(" ;
  sid:100000011; )
35 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "Reflected XSS Detected !"; content: "script" ;
  sid:100000012; )
36 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "Reflected XSS Detected !"; content: "alert" ;
  sid:100000013; )
37 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "Reflected XSS Detected !"; content: "123" ;
  sid:100000014; )
38 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "Reflected XSS Detected !"; content: "321" ;
  sid:100000015; )
39 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "Reflected XSS Detected !"; content: "prompt" ;
  sid:100000016; )
40 alert tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "Reflected XSS Detected !"; content: "href" ;
  sid:100000017; )

```

```

11/24-10:49:02.948504 192.168.73.1:29741 -> 192.168.73.129:1337 [**] [1:100000003:0] SQL Injection Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29741 -> 192.168.73.129:1337
11/24-10:49:02.948504 192.168.73.1:29741 -> 192.168.73.129:1337 [**] [1:100000001:0] SQL Injection Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29741 -> 192.168.73.129:1337
11/24-10:49:28.604175 192.168.73.1:29745 -> 192.168.73.129:1337 [**] [1:10000100:5] HTTP detected broooo ! [**] [Priority: 0] {TCP} 192.168.73.1:29745 -> 192.168.73.129:1337
11/24-10:49:28.604175 192.168.73.1:29745 -> 192.168.73.129:1337 [**] [1:100000003:0] SQL Injection Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29745 -> 192.168.73.129:1337
11/24-10:49:28.604175 192.168.73.1:29745 -> 192.168.73.129:1337 [**] [1:100000001:0] SQL Injection Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29745 -> 192.168.73.129:1337
11/24-10:49:36.094633 192.168.73.1:29747 -> 192.168.73.129:1337 [**] [1:100000012:0] Reflected XSS Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29747 -> 192.168.73.129:1337
11/24-10:49:36.094633 192.168.73.1:29747 -> 192.168.73.129:1337 [**] [1:100000019:0] Reflected XSS Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29747 -> 192.168.73.129:1337
11/24-10:49:36.094633 192.168.73.1:29747 -> 192.168.73.129:1337 [**] [1:100000013:0] Reflected XSS Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29747 -> 192.168.73.129:1337
11/24-10:49:36.094633 192.168.73.1:29747 -> 192.168.73.129:1337 [**] [1:10000100:5] HTTP detected broooo ! [**] [Priority: 0] {TCP} 192.168.73.1:29747 -> 192.168.73.129:1337
11/24-10:49:36.094633 192.168.73.1:29747 -> 192.168.73.129:1337 [**] [1:100000014:0] Reflected XSS Detected ! [**] [Priority: 0] {TCP} 192.168.73.1:29747 -> 192.168.73.129:1337

```

- Now I write the rule to block the IP which using malicious attack to Kali website

```

11 #REJECT
12 #ping
13 reject icmp 192.168.73.129 any -> 192.168.73.128 any (msg:"Reject ping from 129!"; sid:10000155; rev:005;)
14
15 #External
16 reject tcp 192.168.73.1 any -> 192.168.73.129 any (msg: "192.168.73.1 has been blocked !"; sid:100000110; )
17
18

```

- So that the IP of host machine is blocked from our DVWA web site

```

Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Commencing packet processing (pid=3441)
11/24-10:50:32.395534 192.168.73.1:29753 -> 192.168.73.129:1337 [**] [1:100000110:0] 192.168.73.1 has been blocked ! [**] [Priority: 0] {TCP}
192.168.73.1:29753 -> 192.168.73.129:1337
11/24-10:50:32.395534 192.168.73.1:29754 -> 192.168.73.129:1337 [**] [1:100000110:0] 192.168.73.1 has been blocked ! [**] [Priority: 0] {TCP}
192.168.73.1:29754 -> 192.168.73.129:1337
11/24-10:50:32.396445 192.168.73.1:29753 -> 192.168.73.129:1337 [**] [1:100000110:0] 192.168.73.1 has been blocked ! [**] [Priority: 0] {TCP}
192.168.73.1:29753 -> 192.168.73.129:1337
11/24-10:50:32.396445 192.168.73.1:29754 -> 192.168.73.129:1337 [**] [1:100000110:0] 192.168.73.1 has been blocked ! [**] [Priority: 0] {TCP}
192.168.73.1:29754 -> 192.168.73.129:1337
11/24-10:50:32.397654 192.168.73.1:29755 -> 192.168.73.129:1337 [**] [1:100000110:0] 192.168.73.1 has been blocked ! [**] [Priority: 0] {TCP}
192.168.73.1:29755 -> 192.168.73.129:1337
11/24-10:50:32.404674 192.168.73.1:29753 -> 192.168.73.129:1337 [**] [1:100000110:0] 192.168.73.1 has been blocked ! [**] [Priority: 0] {TCP}
192.168.73.1:29753 -> 192.168.73.129:1337
11/24-10:50:34.227717 192.168.73.1:29755 -> 192.168.73.129:1337 [**] [1:100000110:0] 192.168.73.1 has been blocked ! [**] [Priority: 0] {TCP}
192.168.73.1:29755 -> 192.168.73.129:1337
11/24-10:50:34.238396 192.168.73.1:29755 -> 192.168.73.129:1337 [**] [1:100000110:0] 192.168.73.1 has been blocked ! [**] [Priority: 0] {TCP}
192.168.73.1:29755 -> 192.168.73.129:1337
11/24-10:50:34.238668 192.168.73.1:29755 -> 192.168.73.129:1337 [**] [1:100000110:0] 192.168.73.1 has been blocked ! [**] [Priority: 0] {TCP}
192.168.73.1:29755 -> 192.168.73.129:1337

```

- ❖ Thus, we have successfully demo the advance model



## Task

ID	Task	Member
1	<ul style="list-style-type: none"> <li>✓ Technical research the basic and advance demo</li> <li>✓ Make slide and present the demo part</li> <li>✓ Write extra snort's rules</li> </ul>	Trương Văn Rõng
2	<ul style="list-style-type: none"> <li>✓ Technical research the theory of snort</li> <li>✓ Make slide and present the first part</li> <li>✓ Make DVWA website in Kali machine</li> </ul>	Lương Mạnh Tiến
3	<ul style="list-style-type: none"> <li>✓ Research and config the IP of VM machines</li> <li>✓ Make slide and present the second part</li> <li>✓ Install snort and community rules</li> </ul>	Phan Hoàng Nam

## Self Assign

- ✓ Evaluation of the level of completion against the implementation plan is: **100%**
- ✓ Point: **10**

## Answer

Other group don't have any question for our team. I just want to say this is a nice topic. Thank you very much!

63	Group 4	No comment					
64	Group 4	Good presentationno question					
65	Group 4	What a perfect presentation					
66	Group 4	The theory is ok, but the demo is very well-explained and detailed					
67	Group 4	no					
68	Group 4	clearly representation but talk too long					
69	Group 4	I don't have any question for group 4					
70	Group 4	I don't have any question for group4					
71	Group 4	The group presented very well, the group members were all confident and did not look at the slides, the demo was clear and easy to see					

---  
**END**