Project 3: Snort and Syslog

April 3, 2014

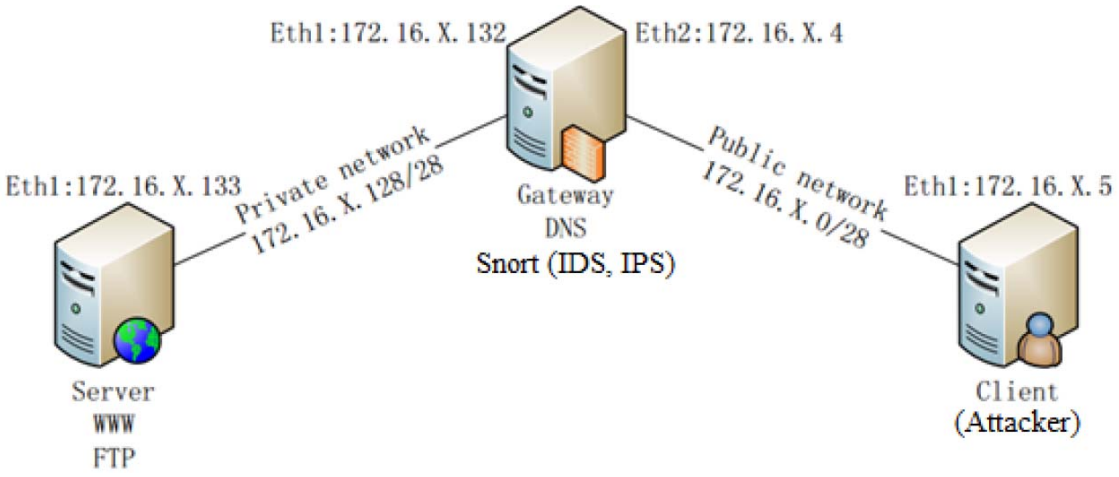
Bing Hao

Project description

The learning objective of this project for students is to gain hands-on experiences with the usage and functionality of Snort and Syslog. After the project, students should know how to set up a network based intrusion detection system using snort to detect and prevent network intrusions, and how to set up a remote syslog server to accept the alert messages generated by Snort. The server student27vmg is supposed be configured as the gateway. The open source IDS and IPS software Snort is installed on the gateway. Logs generated by the gateway should be stored on the server (student27vms).

Network set up of the project

The network topology is illustrated as follows:



(X represents your project number in vLab. Your current IP address should be 172.24.x.x/28)

For the client:

The client's IP address is 172.24.27.6 and its DNS is configured to 172.24.27.133.

For the Server:

The server's IP address is 172.24.27.134 and its DNS is configured to 8.8.8.8.

For the GW:

```
eth0      Link encap:Ethernet  HWaddr fa:16:3e:b6:bd:57
          inet addr:172.24.27.197  Bcast:172.24.27.207  Mask:255.255.255.240
          inet6 addr: fe80::f816:3eff:feb6:bd57/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:75664 errors:0 dropped:0 overruns:0 frame:0
          TX packets:49276 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:76827049 (76.8 MB)  TX bytes:5978729 (5.9 MB)

eth1      Link encap:Ethernet  HWaddr fa:16:3e:59:41:f7
          inet addr:172.24.27.5  Bcast:172.24.27.15  Mask:255.255.255.240
          inet6 addr: fe80::f816:3eff:fe59:41f7/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:60246 errors:0 dropped:0 overruns:0 frame:0
          TX packets:35256 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5515111 (5.5 MB)  TX bytes:25314342 (25.3 MB)

eth2      Link encap:Ethernet  HWaddr fa:16:3e:99:72:a2
          inet addr:172.24.27.133  Bcast:172.24.27.143  Mask:255.255.255.240
          inet6 addr: fe80::f816:3eff:fe99:72a2/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:90060 errors:0 dropped:0 overruns:0 frame:0
          TX packets:34212 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6708065 (6.7 MB)  TX bytes:27100898 (27.1 MB)
```

GW has 3 network cards, 172.24.27.133 will be used as the DNS server and the Gateway for other 2 machines.

Software packages used in the project

- Apache web server
- Bind9 DNS server
- VIM
- Iptables
- Openssh-server
- Snort

Step-by-step project description

1. Install Snort on the server student27vmg using the buildSnort.sh which attached to the Attached files section (Method b). This .sh file is supposed to download the source code of snort and compile & install it to /usr/local/. The snort binary file is under /usr/local/bin.
2. On the server student27vmg , Download and install the rule sets "snortrules-snapshot-2960.tar.gz" from www.snort.org and download the shell script "rules.sh" using "wget" command in the terminal from http://cse468.mobicloud.asu.edu/projects/rules.sh. The rules.sh which had been modified the version to 2960 is attached to the Attached files section (Method b).

```
ubuntu@ubuntu-virtual-machine:~/Downloads$ ls
buildSnort.sh  libpcap-1.4.0          rules.sh
fw.sh          libpcap-1.4.0.tar.gz   snortrules-snapshot-2960.tar.gz
ubuntu@ubuntu-virtual-machine:~/Downloads$
```

3. On the server student27vmg, sudo ./rules.sh
4. On the server student27vmg, adding /usr/local/lib to system's path

```
ubuntu@ubuntu-virtual-machine:~$ export LD_LIBRARY_PATH=/usr/local/lib
```

And
In ~/.bashrc file add following alias:
because sudo doesn't export LD_LIBRARY_PATH for our apps
alias sudo='sudo env LD_LIBRARY_PATH=/usr/local/lib/'

5. On the server student27vmg , link the dynamicrules lib to the correct location:
ln -s /usr/local/snort/so_rules/precompiled/Ubuntu-12-04/i386/2.9.6.0 /usr/local/lib/snort_dynamicrules

6. On the server student27vmg, check the required libs for snort

```
ubuntu@ubuntu-virtual-machine:~$ ldd /usr/local/bin/snort
        linux-gate.so.1 =>  (0xb778a000)
        libdumbnet.so.1 => /usr/lib/libdumbnet.so.1 (0xb7769000)
        libpcre.so.3 => /lib/i386-linux-gnu/libpcre.so.3 (0xb772d000)
        libpcap.so.0.8 => /usr/lib/i386-linux-gnu/libpcap.so.0.8 (0xb76f5000)
        libm.so.6 => /lib/i386-linux-gnu/libm.so.6 (0xb76c9000)
        libdaq.so.2 => /usr/local/lib/libdaq.so.2 (0xb76c3000)
        libdl.so.2 => /lib/i386-linux-gnu/libdl.so.2 (0xb76be000)
        libz.so.1 => /lib/i386-linux-gnu/libz.so.1 (0xb76a8000)
        libpthread.so.0 => /lib/i386-linux-gnu/libpthread.so.0 (0xb768c000)
        libc.so.6 => /lib/i386-linux-gnu/libc.so.6 (0xb74e2000)
        /lib/ld-linux.so.2 (0xb778b000)
```

7. On the server student27vmg, make sure you have installed the required module to perform IPS with NFQ in Snort

```
ubuntu@ubuntu-virtual-machine:~$ snort --daq-dir=/usr/local/lib/daq/ --daq-list
Available DAQ modules:
afpacket(v5): live inline multi unpriv
pcap(v3): readback live multi unpriv
dump(v2): readback live inline multi unpriv
ipfw(v3): live inline multi unpriv
nfq(v7): live inline multi
ubuntu@ubuntu-virtual-machine:~$
```

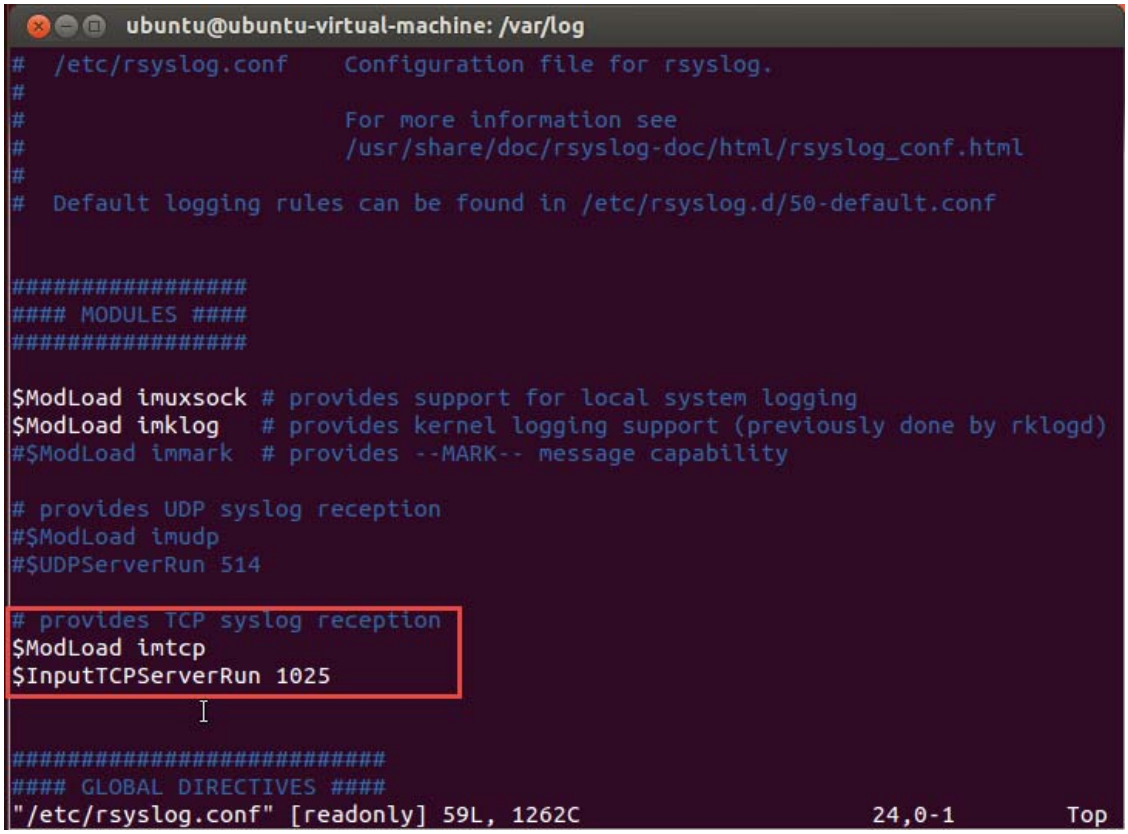
8. Make sure you have Web Server, ssh and syslog (rsyslog) running on the server

   Execute the iptables.sh which was attached to this report in the attached file section.

   Web Server and ssh verification:

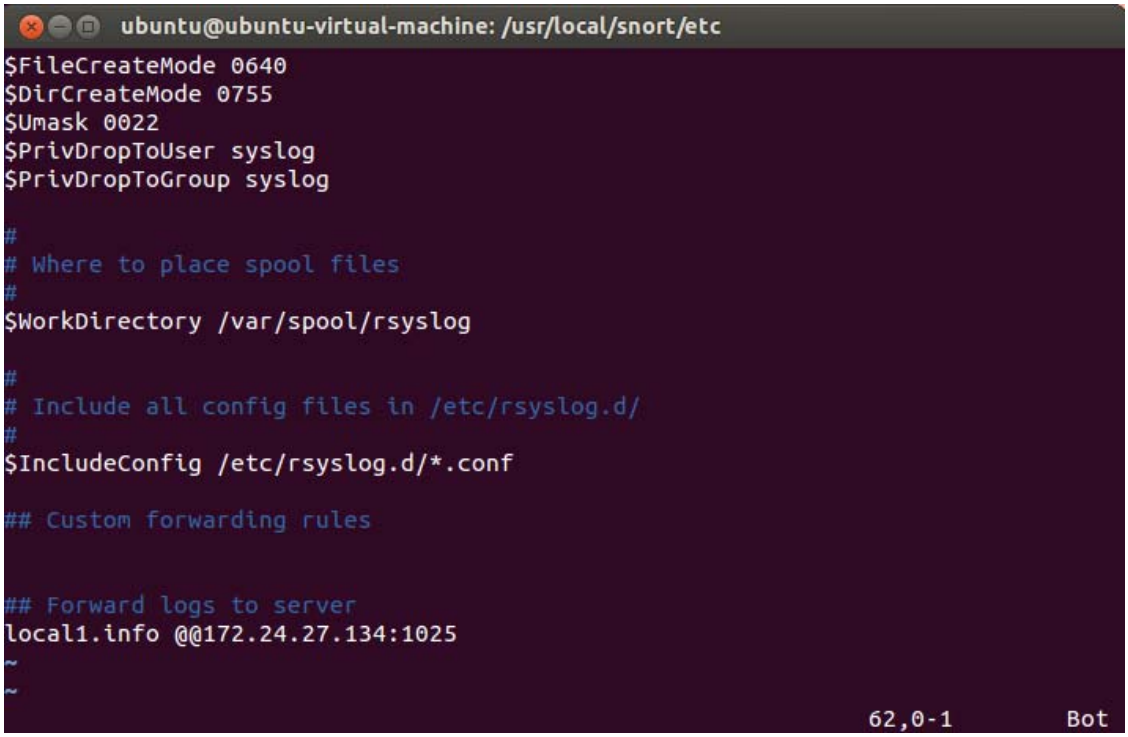
```
ubuntu@ubuntu-virtual-machine:~/Documents$ wget 172.24.27.133:80
--2014-04-07 18:13:08--  http://172.24.27.133/
Connecting to 172.24.27.133:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 213 [text/html]
Saving to: `index.html'

100%[======================================>] 213         --.-K/s   in 0s

2014-04-07 18:13:08 (17.6 MB/s) - `index.html' saved [213/213]

ubuntu@ubuntu-virtual-machine:~/Documents$ ssh 172.24.27.133
The authenticity of host '172.24.27.133 (172.24.27.133)' can't be established.
ECDSA key fingerprint is a0:a4:ed:ed:37:90:45:d7:96:b4:eb:f6:33:5a:8e:7e.
Are you sure you want to continue connecting (yes/no)?
```

On the server student27vms, edit /etc/rsyslog.conf



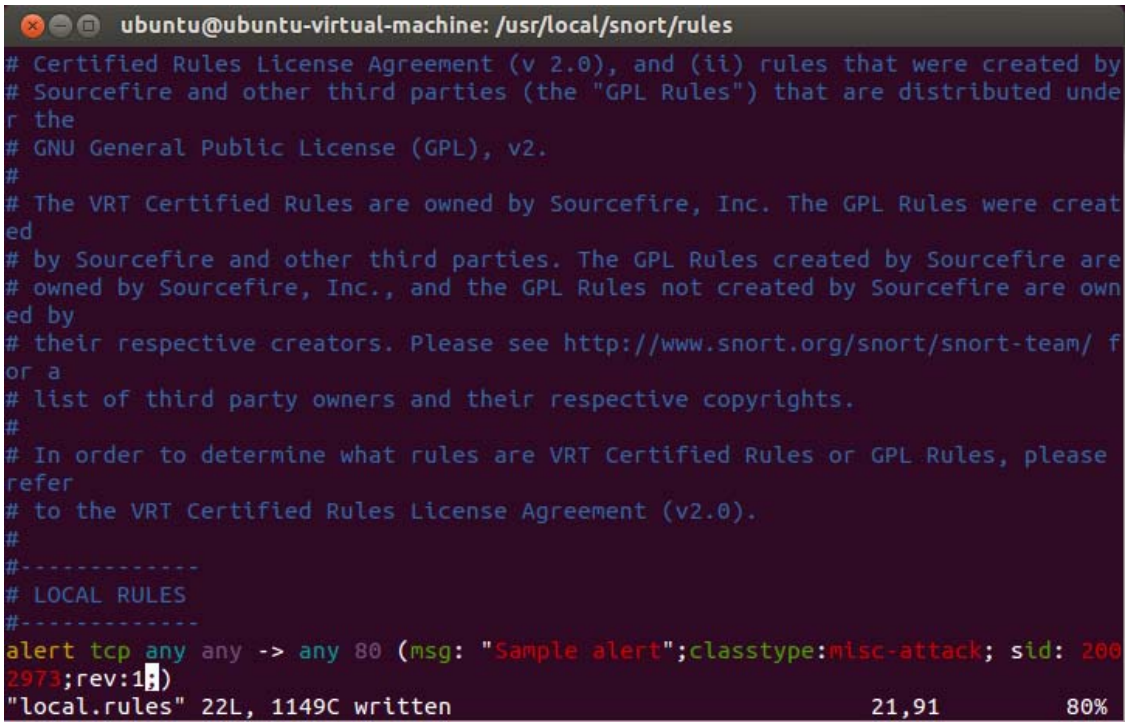
Execute sudo service rsyslog restart to apply the modified conf file.

On the server student27vmg, edit /etc/rsyslog.conf



Execute sudo service rsyslog restart to apply the modified conf file.

On the server student27vmg , testing the Snort by adding a simple rule to /usr/local/snort/rules



Above rule will generate an alert message when any host try to access a web server(on port 80) through the gateway student27vmg.

Execute following command to lunch the snort:

```
ubuntu@ubuntu-virtual-machine:/usr/local/snort/etc$ sudo snort -i eth2 -l /home
/ubuntu/snort -c snort.conf --daq-dir=/usr/local/lib/daq --daq afpacket -A cons
ole
```

Following alert messages were generated in the terminal of the gateway when accessed the web server from client.

```
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
Commencing packet processing (pid=4188)
Decoding Ethernet
04/07-01:08:41.574899  [**] [1:2002973:1] Sample alert [**] [Classification: Mi
sc Attack] [Priority: 2] {TCP} 172.24.27.133:48038 -> 172.24.27.134:80
04/07-01:08:41.577387  [**] [1:2002973:1] Sample alert [**] [Classification: Mi
sc Attack] [Priority: 2] {TCP} 172.24.27.133:48038 -> 172.24.27.134:80
04/07-01:08:41.577516  [**] [1:2002973:1] Sample alert [**] [Classification: Mi
sc Attack] [Priority: 2] {TCP} 172.24.27.133:48038 -> 172.24.27.134:80
04/07-01:08:41.579025  [**] [1:2002973:1] Sample alert [**] [Classification: Mi
sc Attack] [Priority: 2] {TCP} 172.24.27.133:48038 -> 172.24.27.134:80
```

On the server student27vmg, the snort log should be handled by the rsyslog (forward to server student27vms)

ubuntu@ubuntu-virtual-machine: /usr/local/snort/etc

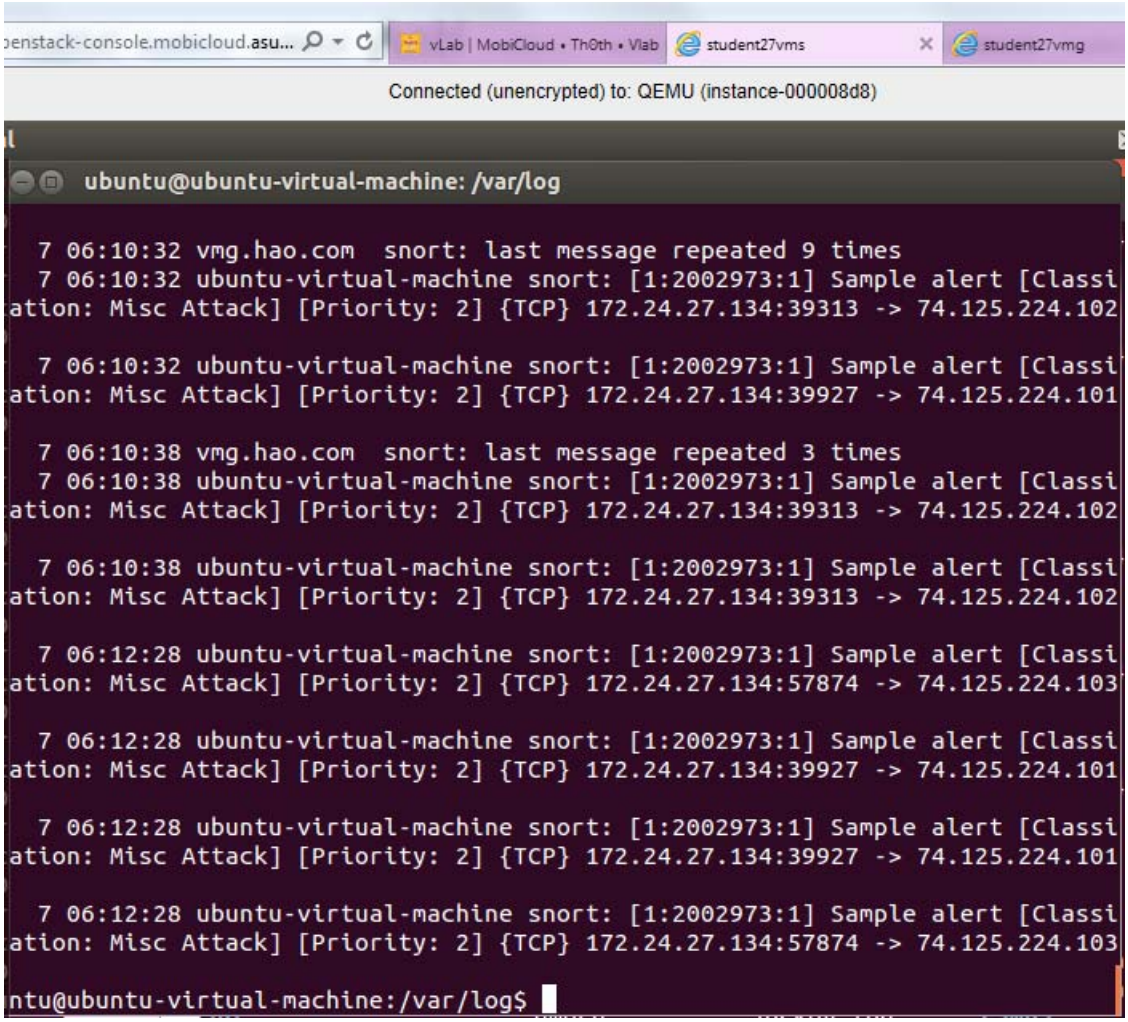
```
# Step #6: Configure output plugins
# For more information, see Snort Manual, Configuring Snort - Output Modules
##################################################

# unified2
# Recommended for most installs
# output unified2: filename merged.log, limit 128, nostamp, mpls_event_types, vl
an_event_types

# Additional configuration for specific types of installs
# output alert_unified2: filename snort.alert, limit 128, nostamp
# output log_unified2: filename snort.log, limit 128, nostamp

# syslog
# output alert_syslog: LOG_AUTH LOG_ALERT
output alert_fast: snort.fast
output alert_syslog:, LOG_LOCAL1 LOG_INFO

# pcap
# output log_tcpdump: tcpdump.log

# metadata reference data.  do not modify these lines
include classification.config
                                                              517,0-1       75%
```

Modify /usr/local/snort/etc/snort.conf to be above.

Execute following command to launch the snort, and trying to access the web site from the client.

```
ubuntu@ubuntu-virtual-machine:/usr/local/snort/etc$ sudo snort -i eth2 -l /var/
log/snort -c snort.conf --daq-dir=/usr/local/lib/daq --daq afpacket
```

On the server student27vms, the /var/log/syslog should be similar as following:



9. On the server student27vmg, for checking any HTTP connection request from attacker to the server, any ssh connection request from attacker to the server and ICMP echo request message with sequence number = 7

Adding following rules in the file /usr/local/snort/rules/local.rules

```
ubuntu@ubuntu-virtual-machine: /usr/local/snort/rules
# their respective creators. Please see http://www.snort.org/snort/snort-team/ f
or a
# list of third party owners and their respective copyrights.
#
# In order to determine what rules are VRT Certified Rules or GPL Rules, please
refer
# to the VRT Certified Rules License Agreement (v2.0).
#
#------------
# LOCAL RULES
#------------
# Any HTTP connection request from attacker to the server
alert tcp any any -> 172.24.27.134 80 (msg: "HTTP from attacker to the server";c
lasstype:misc-attack; sid: 2002975;)
# Any ssh connection request from attacker to the server
alert tcp any any -> 172.24.27.134 22 (msg: "SSH from attacker to the server";cl
asstype:misc-attack; sid: 2002976;)
# ICMP echo request message with sequence number =7
alert icmp any any -> any any (icmp_seq:7; msg: "ICMP echo request sequence numb
er 7";classtype:misc-attack;sid: 2002977;)

~
~
"local.rules" 27L, 1551C written                          26,110        Bot
```

Rules Verification:

```
ubuntu@ubuntu-virtual-machine:/usr/local/snort/etc$ sudo snort -T -c snort.conf
 --daq-dir=/usr/local/lib/daq
```

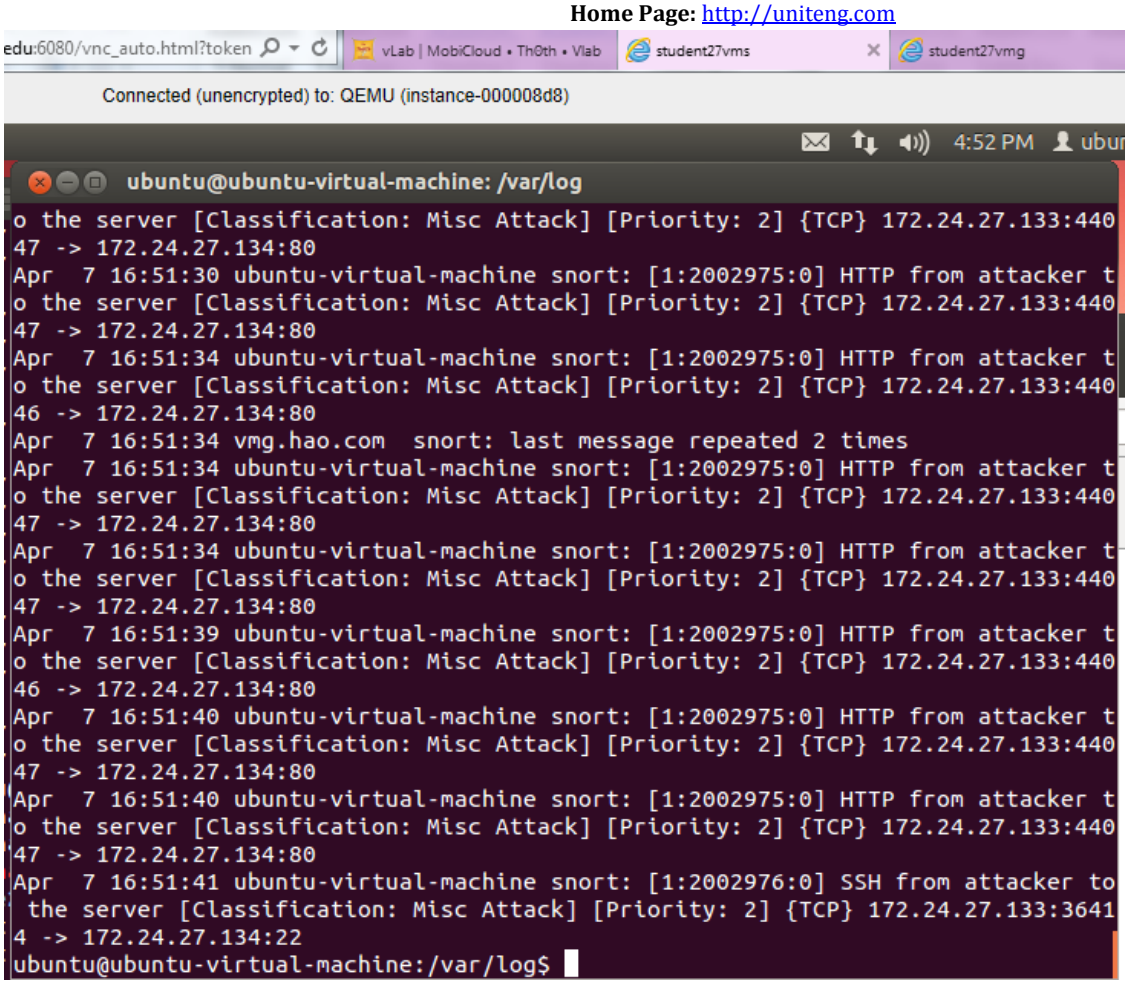
Verification results:

```
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>

Snort successfully validated the configuration!
Snort exiting
```

Launch the snort:

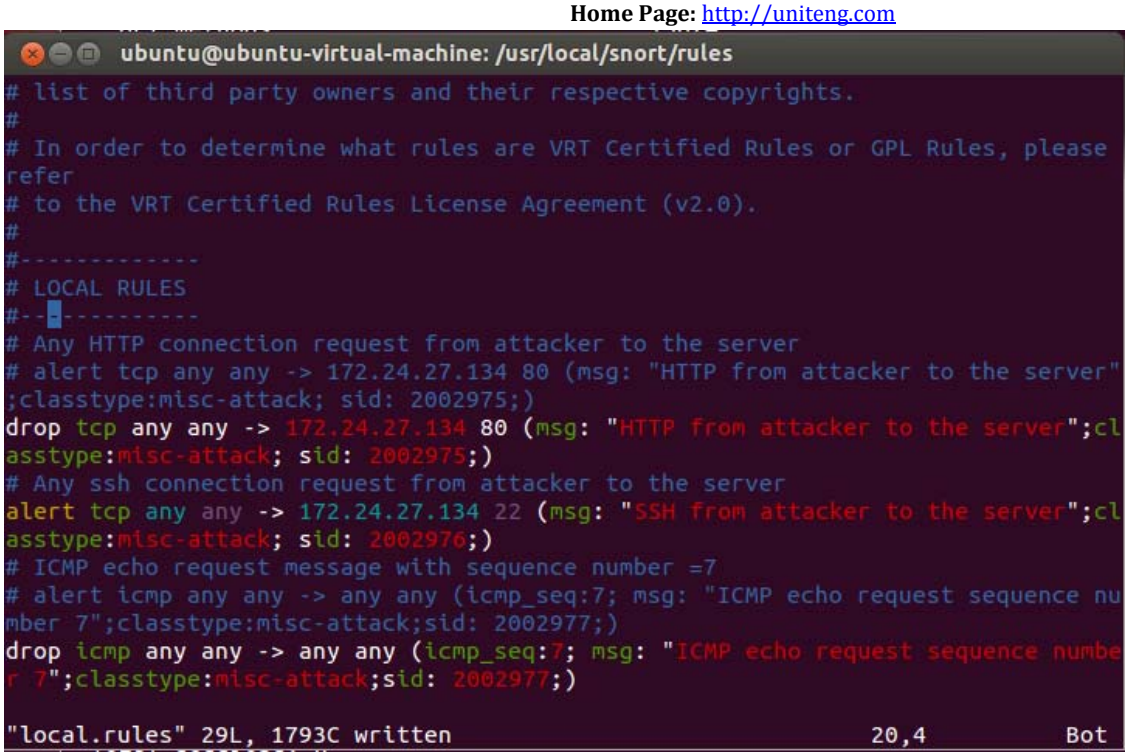
```
ubuntu@ubuntu-virtual-machine:/usr/local/snort/etc$ sudo snort -i eth2 -l /var/
log/snort -c snort.conf --daq-dir=/usr/local/lib/daq --daq afpacket
```

The log on the server after Ping, HTTP accessing and SSH from the client (student27vmc) to the server (student27vms):

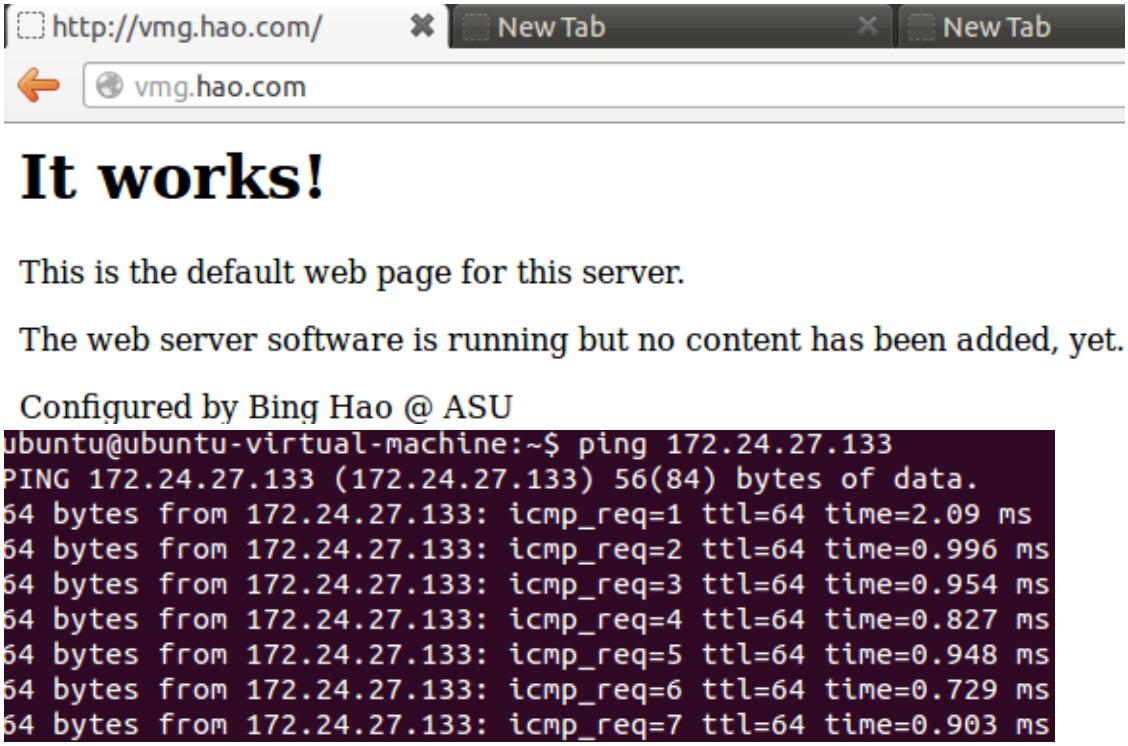


10. For block the ping traffic from the attacker to the server and block HTTP request from the attacker to the server.

    On the server student27vmg, modify rules in the file /usr/local/snort/rules/local.rules to be following, this will drop the messages:



Before launch the snort, we could access the server from the client using HTTP and ICMP:



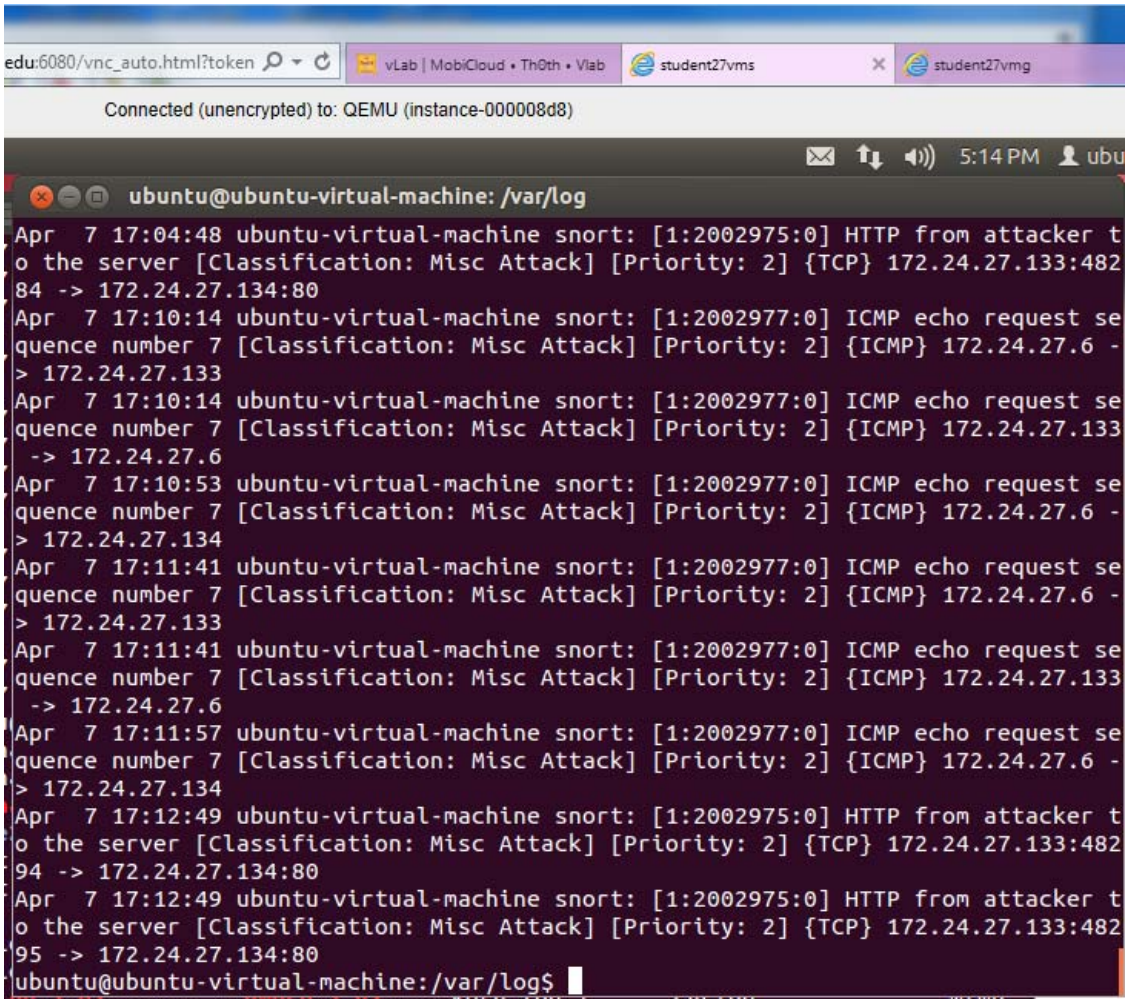

```
ubuntu@ubuntu-virtual-machine:~$ ping 172.24.27.133
PING 172.24.27.133 (172.24.27.133) 56(84) bytes of data.
64 bytes from 172.24.27.133: icmp_req=1 ttl=64 time=2.09 ms
64 bytes from 172.24.27.133: icmp_req=2 ttl=64 time=0.996 ms
64 bytes from 172.24.27.133: icmp_req=3 ttl=64 time=0.954 ms
64 bytes from 172.24.27.133: icmp_req=4 ttl=64 time=0.827 ms
64 bytes from 172.24.27.133: icmp_req=5 ttl=64 time=0.948 ms
64 bytes from 172.24.27.133: icmp_req=6 ttl=64 time=0.729 ms
64 bytes from 172.24.27.133: icmp_req=7 ttl=64 time=0.903 ms
```

Launch the snort with -Q:

```
ubuntu@ubuntu-virtual-machine:/usr/local/snort/etc$ sudo snort -Q -i eth2:eth1
-l /var/log/snort -c snort.conf --daq-dir=/usr/local/lib/daq --daq afpacket
```

After launch the snort, we could not access the server from the client using HTTP and ICMP, the blocked packages had also been logged.

```
ubuntu@ubuntu-virtual-machine:~$ ping 172.24.27.133
PING 172.24.27.133 (172.24.27.133) 56(84) bytes of data.
^Z
[1]+  Stopped                 ping 172.24.27.133
ubuntu@ubuntu-virtual-machine:~$ wget 172.24.27.133:80
--2014-04-07 17:30:23--  http://172.24.27.133/
Connecting to 172.24.27.133:80... ^Z
[2]+  Stopped                 wget 172.24.27.133:80
```



Questions

1. Which approach you're using to install Snort, what's the purpose of each step in the installation script, and where is the Snort binary file located after installation? (Refer to the step 1 and 2 in section 2.1)

   Answer:

   I used approach b to install the snort. Please refer to my comments in section attached files, those comments explain the purpose of each step in the installation script. The Snort binary file located after installation is under /usr/local/bin. Refer Step 1 to Step 7 in the Step-by-step project description section for more details.

2. What's the result you get after performing the command in the step 3 of section 2.1?

   Answer:

   ```
   ubuntu@ubuntu-virtual-machine:~$ snort --daq-dir=/usr/local/lib/daq/ --daq-list
   Available DAQ modules:
   afpacket(v5): live inline multi unpriv
   pcap(v3): readback live multi unpriv
   dump(v2): readback live inline multi unpriv
   ipfw(v3): live inline multi unpriv
   nfq(v7): live inline multi
   ubuntu@ubuntu-virtual-machine:~$
   ```

   Refer Step 7 in the Step-by-step project description section for more details.

3. Describe the detailed steps of how to set up a remote syslog server to accept the log from Snort.

   Answer:

   This question was answered in the Step 8, the Step-by-step project description section.

4. For each task in section 2.2, please describe the detailed steps and the result after the configuration. Please include some major screenshots to proof your configurations are working and satisfy the requirement for each task.

   Answer:

   This question was answered in the Step 9 and Step 10, the Step-by-step project description section.

Conclusion

The project have been finished successfully. The snort and rsyslog have been configured properly.

Attached files

File name: buildSnort.sh

```
#! /bin/bash


# Add snort path to the system
if [ ! "$(grep LD_LIBRARY_PATH /root/.bashrc)" ]
then
  echo "LD_LIBRARY_PATH=/usr/local/lib" >> /root/.bashrc
  echo "export LD_LIBRARY_PATH" >> /root/.bashrc
fi


LD_LIBRARY_PATH=/usr/local/lib
export LD_LIBRARY_PATH


libpcap=libpcap-1.4.0
snort=snort-2.9.6.0
snortDownloadNr=2787
daq=daq-2.0.2
daqDownloadNr=2778


#Update the apt-get database
apt-get update
#Install required software and libs for snort
apt-get install bison flex zlib1g-dev libdumbnet1 libdumbnet-dev libpcre3-dev libnetfilter-queue-dev libnet1-dev libpcap-dev
```

```
# make a new dictionary for snort

mkdir /root/snort


# Quote from http://sourceforge.net/projects/libpcap/:

# "libpcap is a system-independent interface for user-level packet capture.

# libpcap provides a portable framework for low-level network monitoring.

# Applications include network statistics collection, security monitoring,

# network debugging, etc." This library enables snort to capture packets from user space.


cd /root/snort

wget http://www.tcpdump.org/release/$libpcap.tar.gz

tar xfz $libpcap.tar.gz

cd $libpcap

./configure

make

make install


# Quote from http://libdnet.sourceforge.net:

# "libdnet (dumb networking library) provides a simplified, portable

# interface to several low-level networking routines, including

# network address manipulation, network firewalling, network interface lookup

# and manipulation and raw IP packet and Ethernet frame transmission."


cd /root/snort

wget http://libdnet.googlecode.com/files/libdnet-1.12.tgz

tar xfz libdnet-1.12.tgz
```

```
cd libdnet-1.12

./configure "CFLAGS=-fPIC -g -O2"

make

make install


cd /root/snort

wget http://www.snort.org/downloads/$daqDownloadNr

mv $daqDownloadNr $daq.tar.gz

tar xfz $daq.tar.gz

cd $daq

./configure --disable-static --disable-ipq-module

make

make install


snort=snort-2.9.6.0

snortDownloadNr=2787


cd /root/snort

wget http://www.snort.org/downloads/$snortDownloadNr

mv $snortDownloadNr $snort.tar.gz

tar xfz $snort.tar.gz

cd $snort

./configure --with-daq-libraries=/usr/local/lib/daq --disable-static-daq

make

make install
```

File name: rules.sh

```
#! /bin/bash

snortVer=2960

snortDir=/usr/local/snort


#the rules package mast be in current directory

if [ ! -f snortrules-snapshot-$snortVer.tar.gz ]

then

  echo "This scripts expects that snortrules-snapshot-$snortVer.tar.gz is located in the current directory"

  echo "Exits..."

  exit

fi


#extract rules to correct locations

mkdir $snortDir

cp snortrules-snapshot-$snortVer.tar.gz $snortDir

cd $snortDir

tar xfz snortrules-snapshot-$snortVer.tar.gz


wget https://s3.amazonaws.com/snort-org/www/rules/community/community-rules.tar.gz

tar xfz community-rules.tar.gz

ln -s /usr/local/snort/so_rules/precompiled/Ubuntu-12-04/i386/2.9.6.0
/usr/local/lib/snort_dynamicrules


touch $snortDir/rules/white_list.rules

touch $snortDir/rules/black_list.rules
```

Now you should be able to run snort by:

snort --daq nfq --daq-var queue=0 -Q --daq-dir=/usr/local/lib/daq -l /var/log/snort -c /usr/local/snort/etc/snort.conf

File name: iptables.sh

```
#!/bin/sh
iptables -F
iptables -F -t nat

iptables -P FORWARD DROP
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT

#For HTTP
iptables -t nat -A PREROUTING -d 172.24.27.133 -p tcp --dport 80 -j DNAT --to
172.24.27.134:80

iptables -t nat -A POSTROUTING -p tcp --dport 80 -j MASQUERADE

#For SSH
iptables -t nat -A PREROUTING -d 172.24.27.133 -p tcp --dport 22 -j DNAT --to
172.24.27.134:22

iptables -t nat -A POSTROUTING -p tcp --dport 22 -j MASQUERADE


#iptables -A FORWARD -i eth2 -s 172.24.27.134 -p tcp --sport 80 -m state --sta
te ESTABLISHED -j ACCEPT
## iptables -A FORWARD -i eth2 -s 172.24.27.134 -p tcp --sport 80 -j ACCEPT
iptables -A FORWARD -o eth2 -d 172.24.27.133 -p tcp --dport 80 -j ACCEPT
iptables -A FORWARD -o eth2 -d 172.24.27.133 -p tcp --dport 22 -j ACCEPT
iptables -A FORWARD -o eth2 -d 172.24.27.133 -p icmp -j ACCEPT

#comming back
iptables -A FORWARD -m state --state ESTABLISHED -j ACCEPT
iptables -A FORWARD -s 172.24.27.134 -j ACCEPT
iptables -A FORWARD -s 172.24.27.133 -j ACCEPT
#iptables -A FORWARD -i eth2 -s 172.24.27.6 -p tcp --dport 80 -j ACCEPT
#iptables -A FORWARD -s 172.24.27.6 -j ACCEPT
# iptables -A INPUT -i eth1 -s 172.24.27.134 -j DROP
# iptables -A FORWARD -s 172.24.27.6 -d 172.24.27.134 -p tcp --dport 80 -j DRO
P
iptables -A FORWARD -s 172.24.27.6 -p tcp --dport 80 -j ACCEPT
iptables -A FORWARD -s 172.24.27.6 -p tcp --dport 22 -j ACCEPT
```

```
iptables -A OUTPUT -p udp --dport 53 -j ACCEPT
iptables -A INPUT -p udp --sport 53 -j ACCEPT
iptables -A OUTPUT -p tcp --dport 53 -j ACCEPT
iptables -A INPUT -p tcp --sport 53 -j ACCEPT

#ICMP
iptables -t nat -A PREROUTING -p ICMP -i eth2 -j DNAT --to-destination 172.24.
27.134
iptables -t nat -A POSTROUTING -p tcp -j MASQUERADE
```