Real-Time Network Intrusion Detection using SNORT

by Ummed Meel

A Network Intrusion Detection System configured with the updated set of rules can make the network secured against the intrusion attack. Through this article, one can have complete understanding and knowledge of deployment of the SNORT, which is an Open Source Network Detection System, with the real-time detection of an intruder in the network.

A network infrastructure that has several servers and devices is always a target of hackers to breach security to access confidential data. Any unauthorized activity from unknown servers or devices can cause a breach of security, which may lead to loss of confidential data. Network security can be breached as the network intrusion can become a threat through remote or physical access. Only an expert with a deep understanding of hacking and who knows how to prevent the network from the unwanted attacks through various channels can detect the intruders in the network and can make sure the next intrusion attack is prevented easily. A Network Intrusion Detection System configured with the updated set of rules can make the network secured against the intrusion attack. Through this article, one can have complete understanding and knowledge of deployment of the SNORT, which is an Open Source Network Detection System, with the real-time detection of an intruder in the network.

The network is the third layer of the OSI model. When the data arrives at the network layer from the data link layer, it contains source and destination IP address. The network layer always plays an important role as it makes sure (It always make sure if data has reached its final destination or not. If so, this layer formats data into packets and deliver it to the transport layer. Otherwise, the network layer updates the destination address and pushes the frame back to lower layer.) by checking the delivery of the data by its state. A malicious request sent by the intruder always hits the network layer first of the target network. When the intruder tries to enumerate the network, it becomes more important as it detects and triggers the alerts per a predefined set of rules by admin.

Know about SNORT

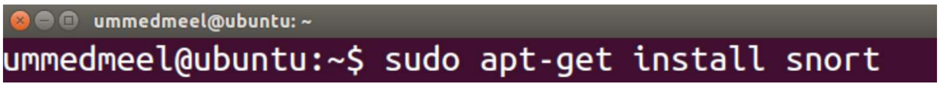
SNORT is the Network Intrusion Detection and Prevention (IDS/IPS) System that is simply an open source tool developed in 1998 by Martin Roesch (a former founder and CTO of SourceFire). Cisco had acquired SourceFire in 2013 and has been developing SNORT since then. SNORT can detect real-time attacks and can protect the network infrastructure from a security breach. It can also detect intrusions from intruders on any website and servers as well. SNORT deeply analyzes the traffic on the network with a predefined set of rules.

SNORT can be easily deployed in the following phases.

Phase 01: Basic Installation

SNORT can be set up on multiple operating systems. We can set up SNORT on Windows, however, for now, we are going to set up SNORT on UBUNTU. Open a terminal in UBUNTU server and run the following command as shown in the image.

Command: "`sudo apt-get install snort`"

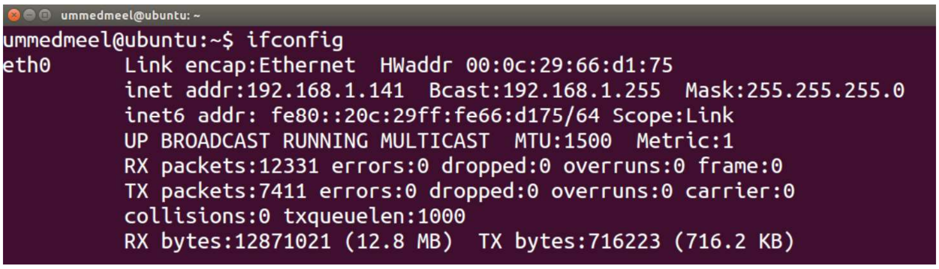


```
ummedmeel@ubuntu:~$ sudo apt-get install snort
```

Fig 01: Install SNORT

Instructions: You would be asked to define the network range to configure SNORT in technical term CIDR for the Home_NET for all the rules defined by the user. To have a completed knowledge of the subnet mask mechanism of the system, you can use the following command.

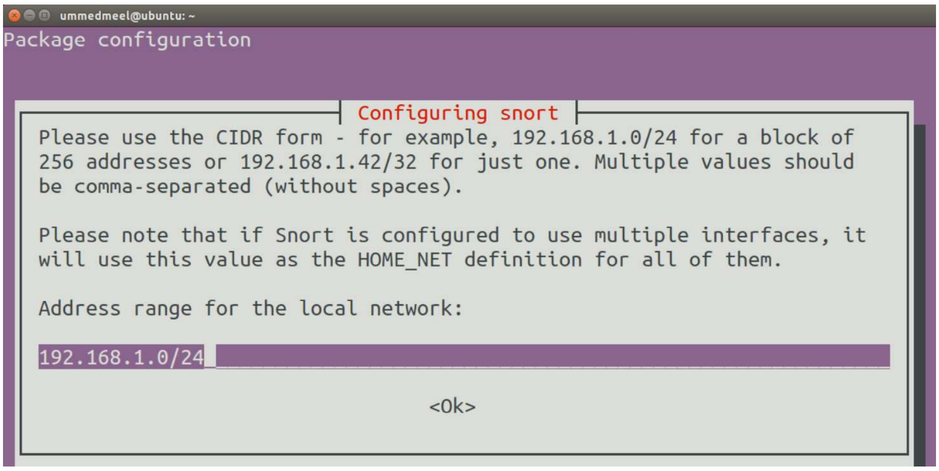
Command: "`Ifconfig`"



```
ummedmeel@ubuntu:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:66:d1:75
          inet addr:192.168.1.141  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe66:d175/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:12331 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7411 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:12871021 (12.8 MB)  TX bytes:716223 (716.2 KB)
```

Fig 02: Ifconfig

We are using 255.255.255.0 subnet mask as shown in the snapshot, doing so we are declaring the network range as 192.168.1.0/24.

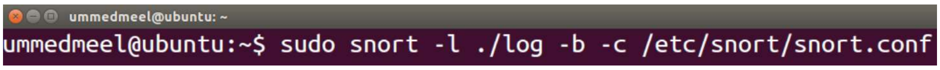


```
Package configuration

                      Configuring snort
 Please use the CIDR form - for example, 192.168.1.0/24 for a block of
 256 addresses or 192.168.1.42/32 for just one. Multiple values should
 be comma-separated (without spaces).

 Please note that if Snort is configured to use multiple interfaces, it
 will use this value as the HOME_NET definition for all of them.

 Address range for the local network:

 192.168.1.0/24

                              <Ok>
```

Fig 03: Subnet Mask

Now run the following command to validate the installation, it would check the entire SNORT configuration with predefined sequence during the installation and would display a successful installation message. Before running this command, you are required to create a log directory to save alert logs (mkdir.log).

Command: "`sudo snort -l ./log -b -c /etc/snort/snort.conf`”



```
ummedmeel@ubuntu:~$ sudo snort -l ./log -b -c /etc/snort/snort.conf
```

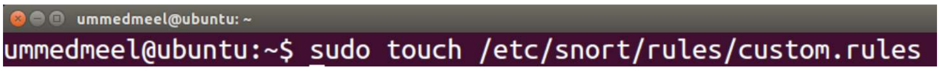
Fig 04: Installation Check

Phase 02: Customization of SNORT Rules

So far, we have learned how to install SNORT and going further we will learn how to customize SNORT with RULES.

Before customizing or setting up the rules on SNORT, an expert is expected to gather knowledge of updated cyber-attacks triggered recently on various platforms, like network infrastructure, web application, and servers, so that rules can be set in SNORT considering any type of threat, breach of security or cyber-attack. Run the following command to create a file designed with intrusion detection rules.

Command: "`sudo touch /etc/snort/rules/custom.rules`"

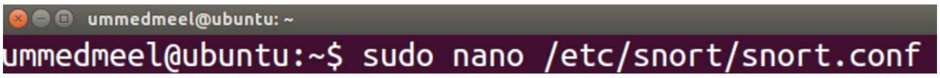


```
ummedmeel@ubuntu:~$ sudo touch /etc/snort/rules/custom.rules
```

Fig 05: Custom rule file creation

Run the following command to make modification in SNORT configuration file.

Command: "`sudo nano /etc/snort/snort.conf`"

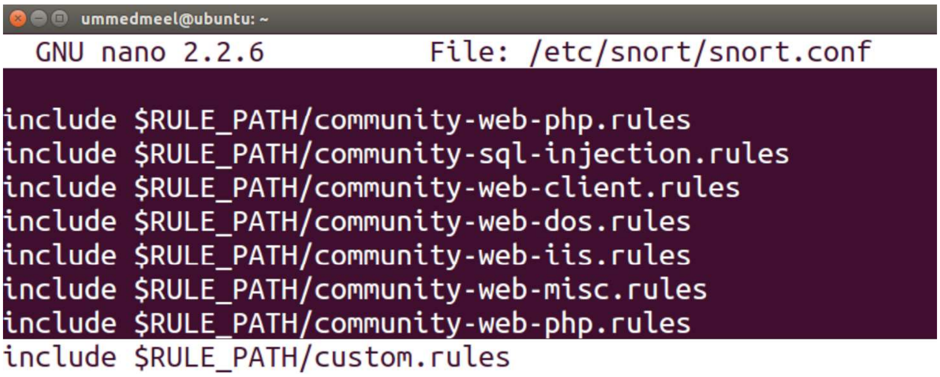


```
ummedmeel@ubuntu:~$ sudo nano /etc/snort/snort.conf
```

Fig 06: Configuration edit

Now in order to add the custom rules in SNORT, we need to include the newly created rule file into the SNORT configuration file. To save the configuration file, use “CTRL+X” followed by “Y” key and hit the enter key.

Add new line as ”include $RULE_PATH/custom.rules"



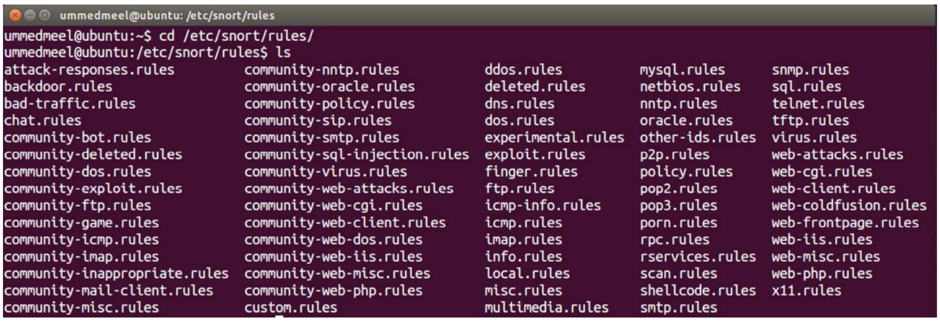
ummedmeel@ubuntu: ~

```
  GNU nano 2.2.6          File: /etc/snort/snort.conf

include $RULE_PATH/community-web-php.rules
include $RULE_PATH/community-sql-injection.rules
include $RULE_PATH/community-web-client.rules
include $RULE_PATH/community-web-dos.rules
include $RULE_PATH/community-web-iis.rules
include $RULE_PATH/community-web-misc.rules
include $RULE_PATH/community-web-php.rules
include $RULE_PATH/custom.rules
```

Fig 07: Add new rule file

Before adding a new rule, if you want to check and update the default set of rules in SNORT, you need to go to rules directory by using the following command.

Command: "cd /etc/snort/rules"



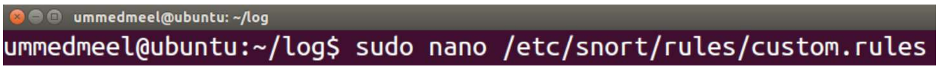
ummedmeel@ubuntu: /etc/snort/rules

```
ummedmeel@ubuntu:~$ cd /etc/snort/rules/
ummedmeel@ubuntu:/etc/snort/rules$ ls
attack-responses.rules         community-nntp.rules           ddos.rules          mysql.rules      snmp.rules
backdoor.rules                 community-oracle.rules         deleted.rules       netbios.rules    sql.rules
bad-traffic.rules              community-policy.rules         dns.rules           nntp.rules       telnet.rules
chat.rules                     community-sip.rules            dos.rules           oracle.rules     tftp.rules
community-bot.rules            community-smtp.rules           experimental.rules  other-ids.rules  virus.rules
community-deleted.rules        community-sql-injection.rules  exploit.rules       p2p.rules        web-attacks.rules
community-dos.rules            community-virus.rules          finger.rules        policy.rules     web-cgi.rules
community-exploit.rules        community-web-attacks.rules    ftp.rules           pop2.rules       web-client.rules
community-ftp.rules            community-web-cgi.rules        icmp-info.rules     pop3.rules       web-coldfusion.rules
community-game.rules           community-web-client.rules     icmp.rules          porn.rules       web-frontpage.rules
community-icmp.rules           community-web-dos.rules        imap.rules          rpc.rules        web-iis.rules
community-imap.rules           community-web-iis.rules        info.rules          rservices.rules  web-misc.rules
community-inappropriate.rules  community-web-misc.rules       local.rules         scan.rules       web-php.rules
community-mail-client.rules    community-web-php.rules        misc.rules          shellcode.rules  x11.rules
community-misc.rules           custom.rules                   multimedia.rules    smtp.rules
```

Fig 08: SNORT default rules

In order to customize the custom file with a new set of rules, you need to open the custom file "custom. rules". Run the following command.

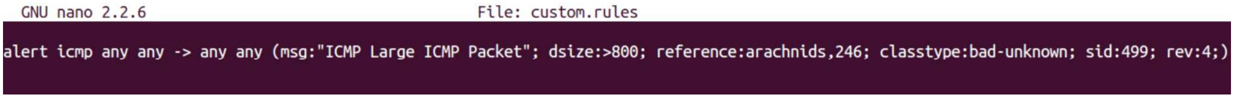
Command: "sudo nano etc/snort/rules/custom.rules"



```
ummedmeel@ubuntu:~/log$ sudo nano /etc/snort/rules/custom.rules
```

Fig 09: Edit custom rule file

One can add a new set of rules according to the latest vulnerability identified. Here we are trying with a simple rule to detect the flow of large ICMP packets in the network. In custom rule, as mentioned in the snapshot first, "any any" where first "any" denotes the attacker's IP address and second "any" denotes the port number of the attacker respectively. Similarly, the second "any any" after the arrow where first "any" denotes the victim's IP address and second "any" denotes the victim's port number respectively. A user can also add a customized alert message for clear understanding during the real-time monitoring. Expert or Tech also needs to assign the unique identification numbers (SID) to the app package and size of ICMP packets.



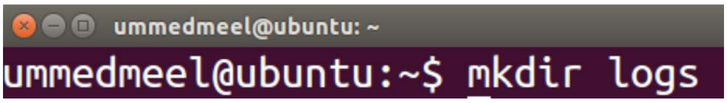
```
  GNU nano 2.2.6                          File: custom.rules

alert icmp any any -> any any (msg:"ICMP Large ICMP Packet"; dsize:>800; reference:arachnids,246; classtype:bad-unknown; sid:499; rev:4;)
```

Fig10: Add new rule

Phase 03: Intrusion Log Storage

So far, we have learned the deployment and customization of rules of SNORT, going forward we will learn how logs get stored in SNORT.

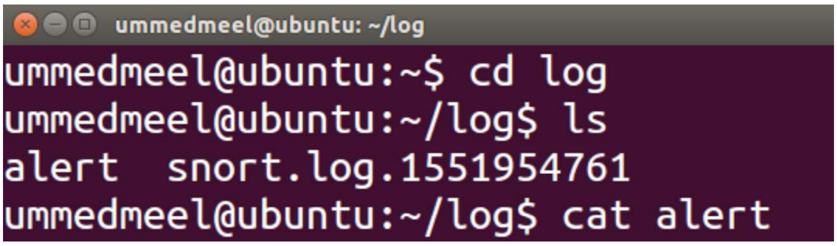
Every time it is not possible to monitor the traffic on the network on a real-time basis to detect the cyber intrusion, that's why experts need to create a folder to save logs of activities that take place during an attack in SNORT. Run “mkdir logs” command to create a directory.



```
ummedmeel@ubuntu:~$ mkdir logs
```

Fig 11: Create log directory

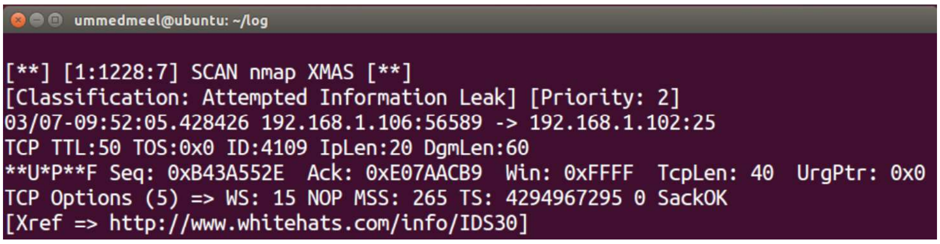
Move to the log folder and check that SNORT has started saving alert logs successfully.



```
ummedmeel@ubuntu:~$ cd log
ummedmeel@ubuntu:~/log$ ls
alert  snort.log.1551954761
ummedmeel@ubuntu:~/log$ cat alert
```

Fig 12: Check log directory

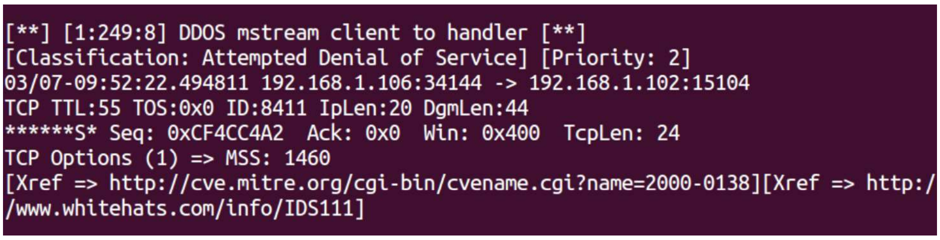
For example, if an intruder tried to scan the network or server using NMAP or other network mapping tools, SNORT will save this as an alert to the logs folder.



```
[**] [1:1228:7] SCAN nmap XMAS [**]
[Classification: Attempted Information Leak] [Priority: 2]
03/07-09:52:05.428426 192.168.1.106:56589 -> 192.168.1.102:25
TCP TTL:50 TOS:0x0 ID:4109 IpLen:20 DgmLen:60
**U*P**F Seq: 0xB43A552E  Ack: 0xE07AACB9  Win: 0xFFFF  TcpLen: 40  UrgPtr: 0x0
TCP Options (5) => WS: 15 NOP MSS: 265 TS: 4294967295 0 SackOK
[Xref => http://www.whitehats.com/info/IDS30]
```

Fig 13: NMAP alert message

Similarly, if any user tries to attack the server or network with DDoS that has SNORT configured, it would be detected and the same activity would be logged with an alert message in the log file as shown below.



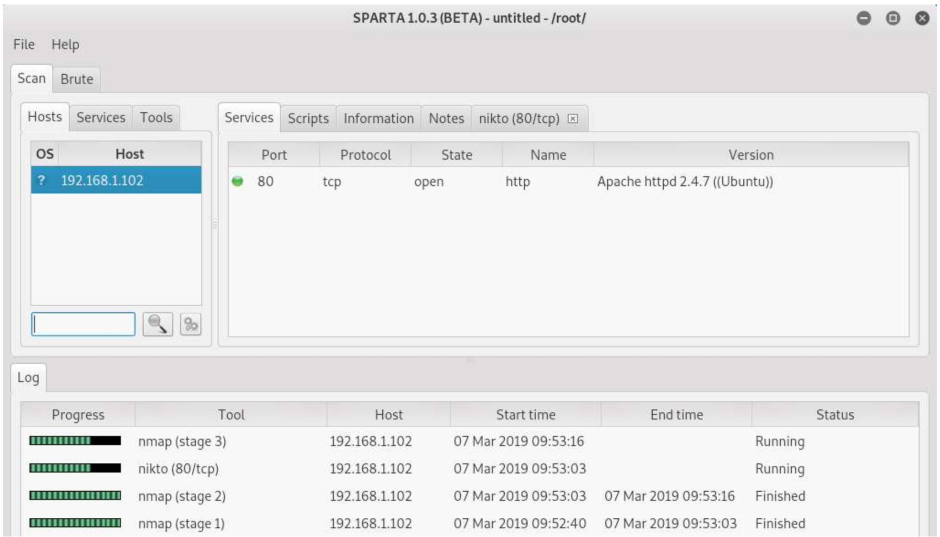
```
[**] [1:249:8] DDOS mstream client to handler [**]
[Classification: Attempted Denial of Service] [Priority: 2]
03/07-09:52:22.494811 192.168.1.106:34144 -> 192.168.1.102:15104
TCP TTL:55 TOS:0x0 ID:8411 IpLen:20 DgmLen:44
******S* Seq: 0xCF4CC4A2  Ack: 0x0  Win: 0x400  TcpLen: 24
TCP Options (1) => MSS: 1460
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0138][Xref => http:/
/www.whitehats.com/info/IDS111]
```

Fig 14: DDoS alert message

Phase 04: Attack using vulnerability scanner tool (SPARTA)

Now we are done with the deployment, customization and setting path for logs in SNORT. Going forward we will test SNORT potential with a test attack using a tool called SPARTA.

Sparta is a vulnerability scanning tool that scans the open ports and runs the service of the target server or network completely. We would be initiating a network scan and vulnerability finder attack on the target server. Later on, Nikto scan open ports discovered by network mapping tool again for the vulnerability on the running applications. During the network mapping and vulnerability scanning, SPARTA is going to hit the target server actively with tons of malicious requests.



OS	Host
?	192.168.1.102

Port	Protocol	State	Name	Version
80	tcp	open	http	Apache httpd 2.4.7 ((Ubuntu))

Progress	Tool	Host	Start time	End time	Status
	nmap (stage 3)	192.168.1.102	07 Mar 2019 09:53:16		Running
	nikto (80/tcp)	192.168.1.102	07 Mar 2019 09:53:03		Running
	nmap (stage 2)	192.168.1.102	07 Mar 2019 09:53:03	07 Mar 2019 09:53:16	Finished
	nmap (stage 1)	192.168.1.102	07 Mar 2019 09:52:40	07 Mar 2019 09:53:03	Finished

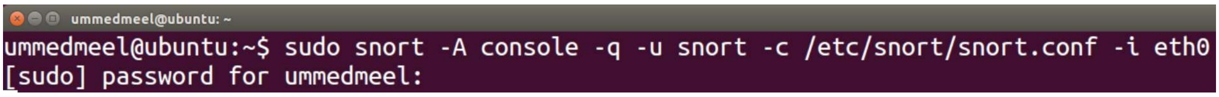
Fig 15: SPARTA scanning tool

Phase 05: Live Logs

During the test attack on SNORT configured network, we can monitor the live logs. Previously, we created a log folder that saves all the possible alerts detected in SNORT. In order to monitor the real-time traffic on a network configured with SNORT, please run the following command.

Command: "`Sudo snort -A console -q -u snort -c /etc/snort/snort.conf -i eth0`"

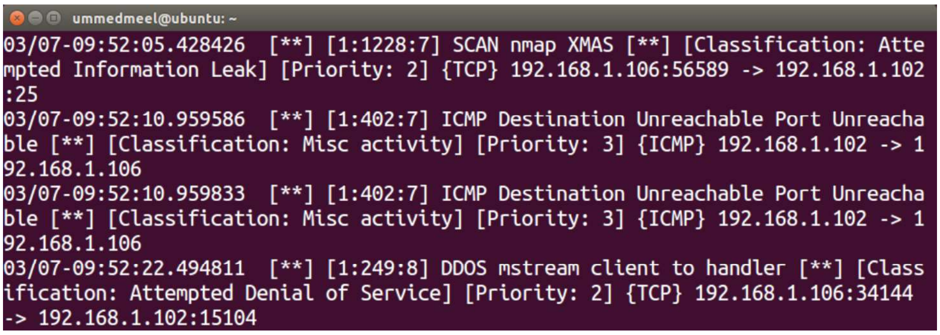
Note: In the above-mentioned command "eth0" defines the adapter of the physical machine through which we are intercepting network traffic.



```
ummedmeel@ubuntu:~$ sudo snort -A console -q -u snort -c /etc/snort/snort.conf -i eth0
[sudo] password for ummedmeel:
```

Fig 16: Real-time log monitoring command

After running the previous command, the network admin or user needs to keep his eyes on the terminal continuously. Every malicious request sent by the intruder could be published here real time with the alert message and attack category. Network traffic intercepted at the ethernet adapter first goes through the alert rules defined by the SNORT and user, after that, if any malicious request or scanning attack request is found, then SNORT displays an alert message live on the terminal and also saves a copy for the user for future reference.



```
03/07-09:52:05.428426  [**] [1:1228:7] SCAN nmap XMAS [**] [Classification: Atte
mpted Information Leak] [Priority: 2] {TCP} 192.168.1.106:56589 -> 192.168.1.102
:25
03/07-09:52:10.959586  [**] [1:402:7] ICMP Destination Unreachable Port Unreacha
ble [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.1.102 -> 1
92.168.1.106
03/07-09:52:10.959833  [**] [1:402:7] ICMP Destination Unreachable Port Unreacha
ble [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.1.102 -> 1
92.168.1.106
03/07-09:52:22.494811  [**] [1:249:8] DDOS mstream client to handler [**] [Class
ification: Attempted Denial of Service] [Priority: 2] {TCP} 192.168.1.106:34144
-> 192.168.1.102:15104
```

Fig 17: Real-time attack log

After real-time detection of cyber intrusion, the user can prevent the intruder from getting into the network. In the case of a DDoS attack on the network, the admin can directly block the attacker's IP address. Now we can see clearly that SNORT has detected a real-time intrusion attack from 192.168.1.106 IP to the 192.168.1.102 IP address and displayed a message as "Attempt to Information Leak".

Conclusion:

So far we should have the complete knowledge of SNORT and its requirement for the security of network infrastructure, Web-application, and Servers. SNORT, which is an open source software, becomes very essential for security purposes as it's easy to configure and can be used on multiple operating systems with few sets of commands. A tech with knowledge of hacking can design the set of rules and protect the infrastructure of any organization using SNORT.

About the Author

Designation: Cyber Expert (Police and Defence Trainer)

Education: B Tech (ECE), CEH (EC-Council), Diploma in Cyber Law, LLB (Pursuing)

Ummed is closely associated with Indian Police, Air Force, BSF and higher defence authorities in India for the last 5+ years. Ummed has conducted 100+trainings and workshops for Indian Police of Delhi, Uttar Pradesh, Rajasthan, Haryana and Himachal police, etc., and trained more than 8,000 police and other LEA's officers. He has 5+ years of experience in Cyber Security, Vulnerability Assessment and Penetration Testing, Cybercrime investigation, Digital Evidence seizure and Digital Forensics. He has been interviewed by several news channels, newspapers and magazines for Cyber Crime Safety and forensics. He has also conducted 50+seminars for government/non-government organizations, school, colleges and other law enforcement agencies on Cybercrime awareness.

Contact: +91 8010 6363 59

Email: ummedmeel@gmail.com

Website: www.ummedcyber.com

eForensics Magazine

MAGAZINE

Cuckoo Sandbox and Malware Analysis

A view into Cuckoo's internals and emulation flow

Extracting Sample Information using Cuckoo API with Python

The First Responder CSIRT on a SECURE Drive

VOL.08 NO.03
ISSUE 03/2019, (88) MARCH
ISSN 2300 6986