Lab 12 (Week 13)
Operational Security – Network Scanner, Firewall, and IDS

CAN201

Dr. Gordon Boateng

Dr. Fei Cheng

Outline

- Scanner
- Firewall
- IDS
- A test run involving nmap, ufw and snort
- Hands-on practice

Scanner

- A scanner can disclose a target's service as well as the corresponding open ports via sending probing traffic against the target (which is a node on the Internet).
  - Recall the difference between a scanner (Nmap) and a sniffer (Wireshark).
- Nmap is a free and open-source utility for network discovery (scanner).

Scanner

- How to install nmap on Ubuntu?
  - **sudo apt-get install nmap**
- How to use nmap?
  - **nmap [scan types] <options> {target specification}**
  - target specification: hostnames, IP addresses, networks, etc.
  - scan types + options:
    - scan techniques:
      - ✓ -sT for TCP Connect Scan
      - ✓ -sS for SYN Scan
      - ✓ -sU for UDP Scan
      - ✓ -sV for service/version detection
      - ✓ -O for OS detection

Firewall

- Firewall on Ubuntu
  - iptables (https://www.netfilter.org/)
  - ufw/gufw (https://wiki.ubuntu.com/UncomplicatedFirewall)
  - firewall builder (http://fwbuilder.sourceforge.net/)
  - etc.
- How to install ufw on Ubuntu
  - **sudo apt-get install ufw**

IDS

- Intrusion Detection System (IDS)
  - Signature-based IDS
    - use well-known signature to detect attack
    - Snort (https://www.snort.org/)
  - Anomaly-based IDS
    - use normal behavior reference profile to detect anomaly
    - Zeek (https://zeek.org/)

IDS

- How to install Snort on Ubuntu
  - **sudo apt-get install snort**
  - To test if Snort is installed, type the command “**snort -V**”.

```
root@can201-VirtualBox:~# snort -V

   ,,_     -*> Snort! <*-
  o"  )~   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
```

A test run involving nmap, ufw and snort

1) Start two VMs, i.e., VM1 and VM2, (e.g., NAT network, or Host-only network). For e.g., VM 1 uses IP address 192.168.56.101, and VM 2 uses IP address 10.0.2.15 .
2) On VM 1, open a terminal and create a simple http server by typing the command "**python3 -m http.server --bind 192.168.56.101 80**" .

```
root@can201-VirtualBox:~# python3 -m http.server --bind 192.168.56.101 80
Serving HTTP on 192.168.56.101 port 80 (http://192.168.56.101:80/) ...
```

3) On VM 2, open a terminal and test if the http server on VM 1 is working correctly by using the following command "**wget -o - 192.168.56.101**" .

If port is in use:

sudo lsof –t –i :80

sudo kill <pid>

```
root@can201-VirtualBox:~# wget -o - 192.168.56.101
--2025-12-05 14:02:14--  http://192.168.56.101/
Connecting to 192.168.56.101:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 860 [text/html]
Saving to: 'index.html'

     0K                                                       100% 1.01M=0.001s

2025-12-05 14:02:14 (1.01 MB/s) - 'index.html' saved [860/860]
```

A test run involving nmap, ufw and snort

4) On VM 2, scan the VM 1 by using "**nmap -sT -sV 192.168.56.101**".

```
root@can201-VirtualBox:~# nmap -sT -sV 192.168.56.101
Starting Nmap 7.80 ( https://nmap.org ) at 2025-12-05 14:02 CST
Nmap scan report for 192.168.56.101
Host is up (0.0020s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE VERSION
80/tcp open  http    SimpleHTTPServer 0.6 (Python 3.8.10)

Service detection performed. Please report any incorrect results at https://nma
p.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.14 seconds
```

Caution: Do not use nmap to scan public network (illegal).

A test run involving nmap, ufw and snort

5) On VM 1, open a new terminal, and deny http traffic by using the ufw: "**ufw deny from 10.0.2.4 to 192.168.56.101 port 80**" and "**ufw enable**".

```
root@can201-VirtualBox:~# ufw deny from 10.0.2.15 to 192.168.56.101 port 80
Skipping adding existing rule
```

6) On VM 2, try this command again "**wget -o - 192.168.56.101**"

```
root@can201-VirtualBox:~# wget -o - 192.168.56.101
--2025-12-05 14:05:08--  http://192.168.56.101/
Connecting to 192.168.56.101:80... connected.
```

7) Go back to VM 1, type the command "**ufw disable**" to stop the firewall.

```
root@can201-VirtualBox:~# ufw disable
Firewall stopped and disabled on system startup
```

A test run involving nmap, ufw and snort

8) On VM 1, open a new terminal, and type the following command to edit the snort rule file "**vim (gedit) /etc/snort/rules/local.rules**".

```
root@can201-VirtualBox:~# vim /etc/snort/rules/local.rules
```

9) In the vim editor, press "i" button to go to the edit mode (it will show "insert" at the bottom on the terminal), and thereafter you can insert a rule to the snort rule file.

Insert the following rule:

alert tcp any any -> 10.0.2.9 80

(msg: "HTTP event"; sid: 1000009;)

A test run involving nmap, ufw and snort

10) With the vim editor, first, press the “Esc” button on your keyboard; second, press “shift” + “:” button on your keyboard; third, type “wq” after the “:” and press “enter” button to write the inserted rule into the rule file and quit the vim editor. Right after that, you will go back to the terminal.

A test run involving nmap, ufw and snort

11) On VM 1, type the following command to run the snort IDS: "**snort -l ./ -c /etc/snort/rules/local.rules**".

```
root@can201-VirtualBox:~# snort -l ./ -c /etc/snort/rules/local.rules
Running in IDS mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/rules/local.rules"
Tagged Packet Limit: 256
Log directory = ./
```

12) Then go to VM 2, execute the command "**wget -o - 192.168.56.101**" again to access the http service running on VM 1 (assuming the simple HTTP server is still running on VM 1).

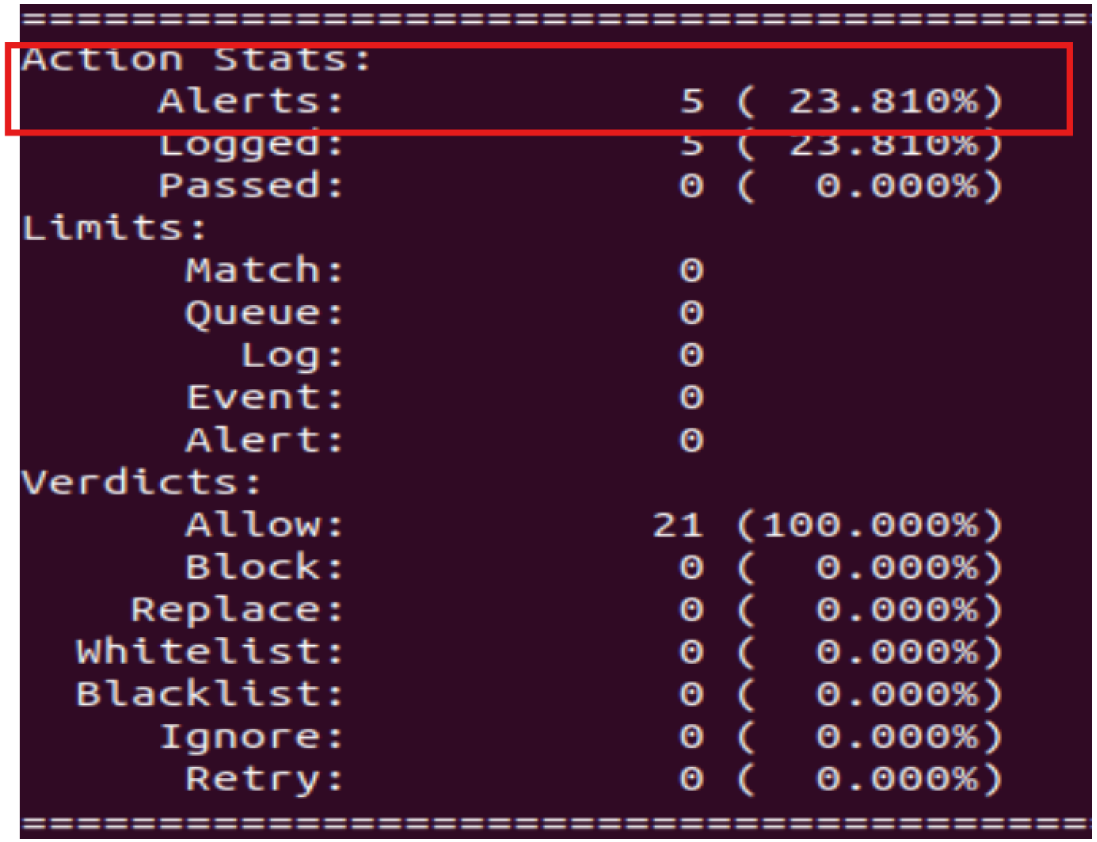
```
root@can201-VirtualBox:~# wget -o - 192.168.56.101
--2025-12-05 14:08:17--  http://192.168.56.101/
Connecting to 192.168.56.101:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 960 [text/html]
Saving to: 'index.html.2'

     0K                                                       100%  236M=0s

2025-12-05 14:08:17 (236 MB/s) - 'index.html.2' saved [960/960]
```

A test run involving nmap, ufw and snort

13) Go to VM 1, stop Snort by pressing the button “Ctrl” + “C”, then you will see it generated some alerts.



A test run involving nmap, ufw and snort

14) On VM 1, you can also view the alerts by read the alert log file which is under the current folder. You can use the command "**ls**" to read detail of the alerts.

```
root@can201-VirtualBox:~# ls
alert  lab99.py  lab9.py  simple_switch_13.py  snap  snort.log.1764914886
root@can201-VirtualBox:~#
```

```
 1 [**] [1:1000009:0] HTTP event [**]
 2 [Priority: 0]
 3 12/05-12:38:14.738211 192.168.56.1:50940 -> 192.168.56.101:80
 4 TCP TTL:128 TOS:0x0 ID:13229 IpLen:20 DgmLen:52 DF
 5 ******S* Seq: 0xEE8D8D8  Ack: 0x0  Win: 0xFFFF  TcpLen: 32
 6 TCP Options (6) => MSS: 1460 NOP WS: 8 NOP NOP SackOK
 7
 8 [**] [1:1000009:0] HTTP event [**]
 9 [Priority: 0]
10 12/05-12:38:14.739198 192.168.56.1:50940 -> 192.168.56.101:80
11 TCP TTL:128 TOS:0x0 ID:13230 IpLen:20 DgmLen:40 DF
12 ***A**** Seq: 0xEE8D8D9  Ack: 0x30606D4B  Win: 0xFF  TcpLen: 20
13
14 [**] [1:1000009:0] HTTP event [**]
15 [Priority: 0]
16 12/05-12:38:14.744554 192.168.56.1:50940 -> 192.168.56.101:80
17 TCP TTL:128 TOS:0x0 ID:13231 IpLen:20 DgmLen:181 DF
18 ***AP*** Seq: 0xEE8D8D9  Ack: 0x30606D4B  Win: 0xFF  TcpLen: 20
19
20 [**] [1:1000009:0] HTTP event [**]
21 [Priority: 0]
22 12/05-12:38:14.748890 192.168.56.1:50940 -> 192.168.56.101:80
23 TCP TTL:128 TOS:0x0 ID:13232 IpLen:20 DgmLen:40 DF
24 ***A**** Seq: 0xEE8D966  Ack: 0x3060776B  Win: 0xFF  TcpLen: 20
25
26 [**] [1:1000009:0] HTTP event [**]
27 [Priority: 0]
28 12/05-12:38:14.771686 192.168.56.1:50940 -> 192.168.56.101:80
```

Hands-on Practice

1) Using Nmap to scan a VM
   - On VM2, run ***nmap –sT –sV <VM1's IP address>***
   - Observe the open ports, detected services, and version information

Questions:

a) Which ports are open on the target machine?

b) What services (e.g., http, ssh, ftp) are running on the ports?

c) Does Nmap correctly detect the version of at least one service?

Hands-on Practice

2) Writing a custom Snort rule that detects ICMP (ping) traffic toward a VM and triggering it

- On VM1, open a Snort rule file: ***sudo vim /etc/snort/rules/local.rules***
- Add this rule: ***alert icmp any any -> <VM1's IP address> any (msg:"ICMP Ping Detected"; sid:10000010;)***
- On VM2, ping VM1: ***ping <VM1's IP address>***

Questions:

a) Did Snort log an alert?

b) Open the Snort alert file (alert) and write down the timestamp of the detected ping.

Thank you!