

# Lab 12 (Week 13)

## Operational Security – Network Scanner, Firewall, and IDS

CAN201

Dr. Wenjun Fan

# Scanner

- A scanner can disclose the 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 scanner and sniffer.
- Nmap is a free and open source utility for network discovery (scanner). It uses raw IP packets in novel ways to determine:
  - what hosts are available on the network
  - what services (application name and version) those hosts are offering
  - what operating systems (and OS versions) they are running
  - what type of packet filters/firewalls are in use
  - dozens of other characteristics



# 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:
      - ✓ -sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
      - ✓ -sU: UDP Scan
    - service/version detection:
      - ✓ -sV: Probe open ports to determine service/version info
    - OS DETECTION:
      - ✓ -O: Enable OS detection



# Scanner

- How to retrieve specific info about the nmap scan options?
  - use command-line “**man nmap**”
  - access to <https://explainshell.com/>



nmap (1) -sT

Network exploration tool and security / port scanner

## **-sT** (TCP connect scan) .

TCP connect scan is the default TCP scan type when SYN scan is not an option. This is the case when a user does not have raw packet privileges or is scanning IPv6 networks. Instead of writing raw packets as most other scan types do, Nmap asks the underlying operating system to establish a connection with the target machine and port by issuing the **connect** system call. This is the same high-level system call that web browsers, P2P clients, and most other network-enabled applications use to establish a connection. It is part of a programming interface known as the Berkeley Sockets API. Rather than read raw packet responses off the wire, Nmap uses this API to obtain status information on each connection attempt.



# 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**
  - if the installation screen ask you to type in the “interface” that snort will listen on, you should type the interface which your VM is using for connecting to the internet. You can check the interface by using the command “**ifconfig**”, in my case, it is “enp0s3” .
  - test if snort is installed, just type command “**snort -V**”.

```
root@bitcoinattacker:/home/wfan# 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 1.8.1
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., A and B, and they locate on the same network (e.g., NAT network, or Host-only network). In my case VM A uses IP address 10.0.2.9, and VM B uses IP address 10.0.2.4 .
- 2) On VM A, open a terminal and create a simple http server by typing the command **“python3 -m http.server --bind 10.0.2.9 80”** .

```
root@bitcoinattacker:/home/wfan# python3 -m http.server --bind 10.0.2.9 80
Serving HTTP on 10.0.2.9 port 80 (http://10.0.2.9:80/) ...
```

- 3) On VM B, open a terminal and test if the http server on VM A is working correctly by using the following command **“wget -o - 10.0.2.9”** .

```
root@controller1:/home/wfan# wget -o - 10.0.2.9
--2021-12-09 14:32:24-- http://10.0.2.9/
Connecting to 10.0.2.9:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1937 (1.9K) [text/html]
Saving to: 'index.html.2'

index.html.2
100% 100% 265M=0s

2021-12-09 14:32:24 (265 MB/s) - 'index.html.2' saved [1937/1937]
```





# A test run involving nmap, ufw and snort

4) On VM B, scan the VM A by using “**nmap -sT -sV 10.0.2.9**”.

```
root@controller1:/home/wfan# nmap -sT -sV 10.0.2.9

Starting Nmap 7.60 ( https://nmap.org ) at 2021-12-09 14:36 UTC
Nmap scan report for 10.0.2.9
Host is up (0.0023s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http      SimpleHTTPServer 0.6 (Python 3.6.9)
MAC Address: 08:00:27:7E:1F:C1 (Oracle VirtualBox Virtual NIC)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.50 seconds
```

Caution: refrain from using nmap to scan public network, which causes ethical issue.



# A test run involving nmap, ufw and snort

- 5) On VM A, open a new terminal, and let's deny http traffic by using the ufw: **"ufw deny from 10.0.2.4 to 10.0.2.9 port 80"** and **"ufw enable"**.

```
root@bitcoinattacker:/home/wfan# ufw deny from 10.0.2.4 to 10.0.2.9 port 80
Rules updated
root@bitcoinattacker:/home/wfan# ufw enable
Firewall is active and enabled on system startup
```

- 6) On VM B, try this command again **"wget -o - 10.0.2.9"**, what will you see?

```
root@controller1:/home/wfan# wget -o - 10.0.2.9
--2021-12-09 14:51:20-- http://10.0.2.9/
Connecting to 10.0.2.9:80...
```

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

```
root@bitcoinattacker:/home/wfan# ufw disable
Firewall stopped and disabled on system startup
```



# A test run involving nmap, ufw and snort

- 8) On VM A, open a new terminal, and type the following command to edit the snort rule file “**vim /etc/snort/rules/local.rules**”.

```
root@bitcoinattacker:/home/wfan# 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.

Here we insert the following rule:

alert tcp any any -> 10.0.2.9 80

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

```
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures.  Put your local
# additions here.
alert tcp any any -> 10.0.2.9 80 (msg: "HTTP event"; sid: 1000009;)
~
~
~
~
~
~
~
~
~
-- INSERT --
```



# 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.

```
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# -----
# LOCAL RULES
# -----
# This file intentionally does not come with signatures. Put your local
# additions here.
alert tcp any any -> 10.0.2.9 80 (msg: "HTTP event"; sid: 1000009;)
~
~
~
~
~
~
~
~
~
```



# A test run involving nmap, ufw and snort

- 11) On VM A, once you go back to the terminal, type the following command to run the snort IDS: **“snort -l ./ -c /etc/snort/rules/local.rules”**.

```
root@bitcoinattacker:/home/wfan# 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 B, execute the command **“wget -o - 10.0.2.9”** again to access the http service running on VM A (assuming the simple HTTP server is still running on VM A).

```
root@controller1:/home/wfan# wget -o - 10.0.2.9
--2021-12-11 10:23:01-- http://10.0.2.9/
Connecting to 10.0.2.9:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2117 (2.1K) [text/html]
Saving to: 'index.html.9'

OK .. 100% 277M=0s

2021-12-11 10:23:01 (277 MB/s) - 'index.html.9' saved [2117/2117]
```



# A test run involving nmap, ufw and snort

13) Go to VM A, stop snort by press button “Ctrl” + “C”, then you will see it generated six alerts. If you used wireshark on VM A to capture the packets when VM B is accessing VM A’s HTTP service, you can verify that there were actually six inbound TCP segments sent to the IP address 10.0.2.9

```
=====
Action Stats:
Alerts:          6 ( 46.154%)
Logged:          6 ( 46.154%)
Passed:          0 (  0.000%)
Limits:
Match:           0
Queue:           0
Log:             0
Event:           0
Alert:           0
Verdicts:
Allow:           13 ( 81.250%)
Block:           0 (  0.000%)
Replace:         0 (  0.000%)
Whitelist:       0 (  0.000%)
Blacklist:       0 (  0.000%)
Ignore:          0 (  0.000%)
Retry:           0 (  0.000%)
=====
Snort exiting
```

ip.dst == 10.0.2.9						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.0.2.4	10.0.2.9	TCP	74	35366 → 80 [SYN] Seq=1068
3	0.000326	10.0.2.4	10.0.2.9	TCP	66	35366 → 80 [ACK] Seq=1068
4	0.000370	10.0.2.4	10.0.2.9	HTTP	138	GET / HTTP/1.1
9	0.001868	10.0.2.4	10.0.2.9	TCP	66	35366 → 80 [ACK] Seq=1068
10	0.001875	10.0.2.4	10.0.2.9	TCP	66	35366 → 80 [ACK] Seq=1068
11	0.019175	10.0.2.4	10.0.2.9	TCP	66	35366 → 80 [FIN, ACK] Seq=

  

▶ Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)	
▶ Ethernet II, Src: PcsCompu_75:3d:42 (08:00:27:75:3d:42), Dst: PcsCompu_7e:1f:c1 (08:00:27:7e:1f:c1)	
▶ Internet Protocol Version 4, Src: 10.0.2.4, Dst: 10.0.2.9	
▶ Transmission Control Protocol, Src Port: 35366, Dst Port: 80, Seq: 1068890812, Len: 0	

  

0000	08 00 27 7e 1f c1 08 00	27 75 3d 42 08 00 45 00	..~... 'u=B..E.
0010	00 3c 45 76 40 00 40 06	dd 39 0a 00 02 04 0a 00	<Ev@.@ -9... ..
0020	02 09 8a 26 00 50 3f b5	fa bc 00 00 00 00 a0 02	..&P? .....
0030	fa f0 bd 71 00 00 02 04	05 b4 04 02 08 0a 98 f2	..q.....
0040	19 b6 00 00 00 00 01 03	03 07	.....

# A test run involving nmap, ufw and snort

- 14) On VM A, you can also view the alerts by read the alert log file which is under the current folder. You can use the command “**less alert**” to read detail of the alerts.

```
root@bitcoinattacker:/home/wfan# ls
Desktop  Downloads  Pictures  Templates  alert
Documents Music      Public   Videos    attack_
root@bitcoinattacker:/home/wfan# less alert
```

```
[**] [1:1000009:0] HTTP event [**]
[Priority: 0]
12/11-10:23:02.114108 10.0.2.4:35370 -> 10.0.2.9:80
TCP TTL:64 TOS:0x0 ID:12409 IpLen:20 DgmLen:60 DF
*****S* Seq: 0x90561E16 Ack: 0x0 Win: 0xFAF0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 2566531822 0 NOP WS: 7

[**] [1:1000009:0] HTTP event [**]
[Priority: 0]
12/11-10:23:02.114478 10.0.2.4:35370 -> 10.0.2.9:80
TCP TTL:64 TOS:0x0 ID:12410 IpLen:20 DgmLen:52 DF
***A*** Seq: 0x90561E17 Ack: 0x69741B23 Win: 0x1F6 TcpLen: 32
TCP Options (3) => NOP NOP TS: 2566531822 3671983928

[**] [1:1000009:0] HTTP event [**]
[Priority: 0]
12/11-10:23:02.114883 10.0.2.4:35370 -> 10.0.2.9:80
:
```



Thank you!

