



第1页 共 页

警示:实验报告如有雷同,雷同各方当次实验成绩均以 0 分计;在规定时间内未上交实验报告的,不得以其他方式补交,当次成绩按 0 分计;实验报告文件以 PDF 格式提交。

院系 计算机学院 班级 <u>软工 3 班</u> 学号 18342075 姓名 米家龙 完成日期: 2020 年 12 月 19 日

网络扫描实验

【实验目的】

- 1. 掌握网络扫描技术的原理。
- 2. 学会使用 Nmap 扫描工具。

【实验环境】

实验主机操作系统: <u>windows 10(搭配wsl)</u> IP地址: <u>172.18.40.111</u> 目标机操作系统: <u>ubuntu mate(树莓派4b,arm64架构)</u> IP地址: <u>172.26.24.231</u> 网络环境: 中山大学校园网(主机有线网,目标机无线网) 。

【实验工具】

Nmap (Network Mapper,网络映射器)是一款开放源代码的网络探测和安全审核的工具。其设计目标是快速地扫描大型网络,也可以扫描单个主机。Nmap 以新颖的方式使用原始 IP 报文来发现网络上的主机及其提供的服务,包括其应用程序名称和版本,这些服务运行的操作系统包括版本信息,它们使用什么类型的报文过滤器/防火墙,以及一些其它功能。虽然 Nmap 通常用于安全审核,也可以利用来做一些日常管理维护的工作,比如查看整个网络的信息,管理服务升级计划,以及监视主机和服务的运行。

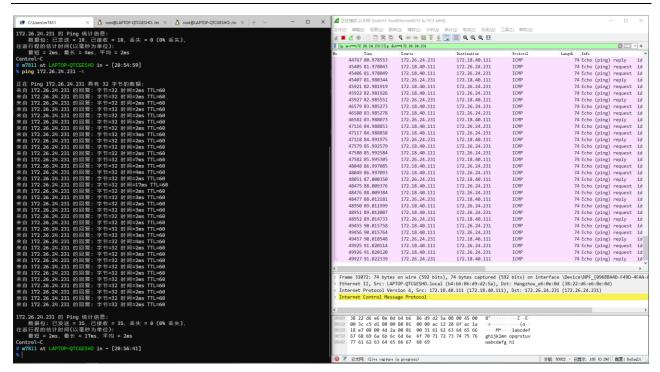
【实验过程】(附带实验截图)

- 1. 测试连通性
 - a) 不开启防火墙, 使用相关命令进行测试
 - i. 使用 ping 命令测试连通性,结果如图,表明连:



Information Security 实验报告

第2页 共 页



ii. 不开启防火墙,使用 nmap 命令测试连通性,终端输出如下:

```
# m7811 at LAPTOP-QTCGESHO in ~ [20:57:12]
% nmap -sP 172.26.24.231
Starting Nmap 7.91 ( https://nmap.org ) at 2020-12-19 20:57 ?D1ú±ê×?ê±??
Nmap scan report for 172.26.24.231
Host is up (0.0030s latency).
Nmap done: 1 IP address (1 host up) scanned in 3.14 seconds
```

查看 wireshark ,发送出现如下包。-sP 选项在默认情况下, 发送一个 ICMP 回声请求和一个 TCP 报文 到 80 端口。如果非特权用户执行,就发送一个 SYN 报文 (用 connect()系统调用)到目标机的 80 端口。 当特权用户扫描局域网上的目标机时,会发送 ARP 请求(-PR),:

No.	Time	Source	Destination	Protocol	Length Info
Г.	292 5.321086	172.18.40.111	172.26.24.231	ICMP	42 Echo (ping) request id=0x4c3f, seq=0/0, ttl=37 (no response found!)
	293 5.321094	172.18.40.111	172.26.24.231	ICMP	42 Echo (ping) request id=0x4c3f, seq=0/0, ttl=37 (reply in 300)
	294 5.324435	172.18.40.111	172.26.24.231	TCP	58 54646 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
	295 5.324441	172.18.40.111	172.26.24.231	TCP	58 [TCP Out-Of-Order] 54646 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
	296 5.324466	172.18.40.111	172.26.24.231	TCP	54 54646 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
	297 5.324468	172.18.40.111	172.26.24.231	TCP	54 [TCP Dup ACK 296#1] 54646 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
	298 5.324480	172.18.40.111	172.26.24.231	ICMP	54 Timestamp request id=0x2b3d, seq=0/0, ttl=43
	299 5.324482	172.18.40.111	172.26.24.231	ICMP	54 Timestamp request id=0x2b3d, seq=0/0, tt1=43
	300 5.325389	172.26.24.231	172.18.40.111	ICMP	60 Echo (ping) reply id=0x4c3f, seq=0/0, ttl=60 (request in 293)
	301 5.335169	172.26.24.231	172.18.40.111	TCP	60 80 → 54646 [RST] Seq=1 Win=0 Len=0
	302 5.335193	172.26.24.231	172.18.40.111	TCP	60 443 → 54646 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
L	303 5.335926	172.26.24.231	172.18.40.111	ICMP	60 Timestamp reply id=0x2b3d, seq=0/0, tt1=60

b) 开启目标机的防火墙,重复①,结果有什么不同?请说明原因。 开启防火墙,目标机使用的是 ubuntu 系统,因此自带了 ufw ,因此使用如下命令开启 ufw



```
Last login: Sat Dec 19 20:12:17 2020 from 172.18.40.111
m@m-desktop:~$ sudo ufw status
[sudo] m 的密码:
状态: 不活动
m@m-desktop:~$ sudo ufw enable
此命令可能会中断目前的 ssh 连接。要继续吗 (y|n)? y
在系统启动时启用和激活防火墙
m@m-desktop:~$ sudo ufw default deny
默认的 incoming 策略更改为 "deny"
(请相应地更新你的防火墙规则)
m@m-desktop:~$ sudo ufw status
状态: 激活
至
                         动作
                                      来自
5700
                        ALLOW
                                    Anywhere
1688
                        ALLOW
                                    Anywhere
22
                        ALLOW
                                    Anywhere
80/tcp
                        ALLOW
                                    Anywhere
5700 (v6)
                        ALLOW
                                    Anywhere (v6)
1688 (v6)
                        ALLOW
                                    Anywhere (v6)
22 (v6)
                        ALLOW
                                    Anywhere (v6)
80/tcp (v6)
                        ALLOW
                                    Anywhere (v6)
```

参考 https://p3terx.com/archives/use-ufw-to-disable-icmp-protocol-access.html, 修改 /etc/ufw/before.rules 文件,使得 ufw 禁止 ping 命令

```
m@m-desktop:~$ nano /etc/ufw/before.rules
m@m-desktop:~$ sudo nano /etc/ufw/before.rules
m@m-desktop:~$ sudo ufw reload
已经重新载入防火墙
m@m-desktop:~$ |
```

修改后的 before.rules 如图:



```
GNU nano 4.8
                                                                                   /etc/ufw/before.rules
 # rules.before
# Rules that should be run before the ufw command line added rules. Custom
# rules should be added to one of these chains:
      ufw-before-input
      ufw-before-forward
# Don't delete these required lines, otherwise there will be errors
*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
:ufw-not-local - [0:0]
# End required lines
# allow all on loopback
 -A ufw-before-input -i lo -j ACCEPT
 -A ufw-before-output -o lo -j ACCEPT
# quickly process packets for which we already have a connection
- A ufw-before-input -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A ufw-before-output -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A ufw-before-forward -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
# drop INVALID packets (logs these in loglevel medium and higher)
-A ufw-before-input -m conntrack --ctstate INVALID -j ufw-logging-deny
-A ufw-before-input -m conntrack --ctstate INVALID -j DROP
# ok icmp codes for INPUT
-A ufw-before-input -p icmp --icmp-type destination-unreachable -j ACCEPT
-A ufw-before-input -p icmp --icmp-type time-exceeded -j ACCEPT
-A ufw-before-input -p icmp --icmp-type parameter-problem -j ACCEPT
# -A ufw-before-input -p icmp --icmp-type echo-request -j ACCEPT
-A ufw-before-input -p icmp --icmp-type echo-request -j DROP
# ok icmp code for FORWARD

-A ufw-before-forward -p icmp --icmp-type destination-unreachable -j ACCEPT

-A ufw-before-forward -p icmp --icmp-type time-exceeded -j ACCEPT

-A ufw-before-forward -p icmp --icmp-type parameter-problem -j ACCEPT

-A ufw-before-forward -p icmp --icmp-type echo-request -j ACCEPT
# allow dhcp client to work
-A ufw-before-input -p udp --sport 67 --dport 68 -j ACCEPT
                                                                                  [已读取 76 行]
                                                                                                          ^J 对齐
^T 拼写检查
                                                                                 ^K 剪切文字
^U 粘贴文字
```

根据要求分别重复步骤 1,结果如图,发现 nmap-sP 能够找到对应的 ip 而 ping 无法验证:

```
# m7811 at LAPTOP-QTCGESHO in ~ [21:33:39]
% ping 172.26.24.231

正在 Ping 172.26.24.231 具有 32 字节的数据:
请求超时。
请求超时。
请求超时。
请求超时。
请求超时。

172.26.24.231 的 Ping 统计信息:
数据包: 已发送 = 4, 已接收 = 0, 丢失 = 4 (100% 丢失),
# m7811 at LAPTOP-QTCGESHO in ~ [21:36:45]
% nmap -sP 172.26.24.231
Starting Nmap 7.91 ( https://nmap.org ) at 2020-12-19 21:36 ?D1ú±ê×?ê±??
Nmap scan report for 172.26.24.231
Host is up (0.0020s latency).
Nmap done: 1 IP address (1 host up) scanned in 2.56 seconds
# m7811 at LAPTOP-QTCGESHO in ~ [21:36:49]
%
```

原因是因为 nmap -sP 选项不仅会发送 icmp 包,还会建立 tcp 连接,从而验证目标 ip 是否存在

c) 测试结果不连通,但实际上是物理连通的,什么原因? 在修改了 ufw 防火墙的配置之后,将原本的 ACCEPT 修改为 DROP 即,将接收到的 icmp 包丢弃,



因此目标机无法对 icmp 来源进行回应,从而导致了测试结果的不连通。

- 2. 对目标主机进行 TCP 端口扫描(目标主机已经开启防火墙)
 - a) 常规扫描 nmap-sT, 扫描结果如下:

```
# m7811 at LAPTOP-OTCGESHO in ~ [21:52:21]
% ping 172.26.24.231
正在 Ping 172.26.24.231 具有 32 字节的数据:
请求超时。
请求超时。
请求超时。
请求超时。
172.26.24.231 的 Ping 统计信息:
    数据包:已发送 = 4,已接收 = 0,丢失 = 4 (100% 丢失),
# m7811 at LAPTOP-QTCGESHO in ~ [21:58:09]
% nmap -sT 172.26.24.231
Starting Nmap 7.91 ( https://nmap.org ) at 2020-12-19 21:59 ?D1ú±êx?ê±??
Nmap scan report for 172.26.24.231
Host is up (0.0029s latency).
Not shown: 998 filtered ports
PORT
        STATE SERVICE
22/tcp
       open ssh
1688/tcp open nsjtp-data
Nmap done: 1 IP address (1 host up) scanned in 44.09 seconds
```

b) 使用 SYN 半扫描 nmap -sS,扫描结果如图:

```
# m7811 at LAPTOP-OTCGESHO in ~ [22:00:21]
% nmap -sS 172.26.24.231
Starting Nmap 7.91 ( https://nmap.org ) at 2020-12-19 22:02 ?D1ú±êx?ê±??
Nmap scan report for 172.26.24.231
Host is up (0.0030s latency).
Not shown: 997 filtered ports
         STATE SERVICE
PORT
         open
22/tcp
                \sim
80/tcp
         closed http
1688/tcp open
                nsjtp-data
Nmap done: 1 IP address (1 host up) scanned in 7.40 seconds
```

- c) 比较上述两次扫描结果差异、扫描所花费的时间。并进行解释。
 - 常规扫描:程序将和目标主机的每个端口都进行**完整的三次握手**。如果成功建立连接,则判定该端口是开放端口。由于在检测每个端口时都需要进行三次握手,所以这种扫描方式比较慢,而且扫描行为很可能被目标主机记录下来。
 - SYN 半扫描: Nmap 将使用 含有 SYN 标志位的数据包进行端口探测。SYN 模式的扫描速度非常好。而且由于这种模式不会进行三次握手,所以是一种十分隐蔽的扫描方式。
 - 如果目标主机回复了 SYN/ACK 包,则说明该端口处 于开放状态;
 - 如果回复的是 RST/ACK 包,则说明这个端口处于关闭状态;
 - 如果没有任何响应或者发送了 ICMP unreachable 信息,则可认为这个端口被屏蔽了。

Information Security 实验报告

第6页 共 页

【实验体会】

在本次实验中,我了解到了网络扫描技术的相关原理,尝试并了解并学习了 nmap 扫描工具的使用方法的技巧,除此之外,还了解到了 ubuntu 系统自带的 ufw 防火墙的相关知识,以及相关的配置文件。除此之外,再次使用了 wireshark 软件,复习了相关的过滤规则和操作。