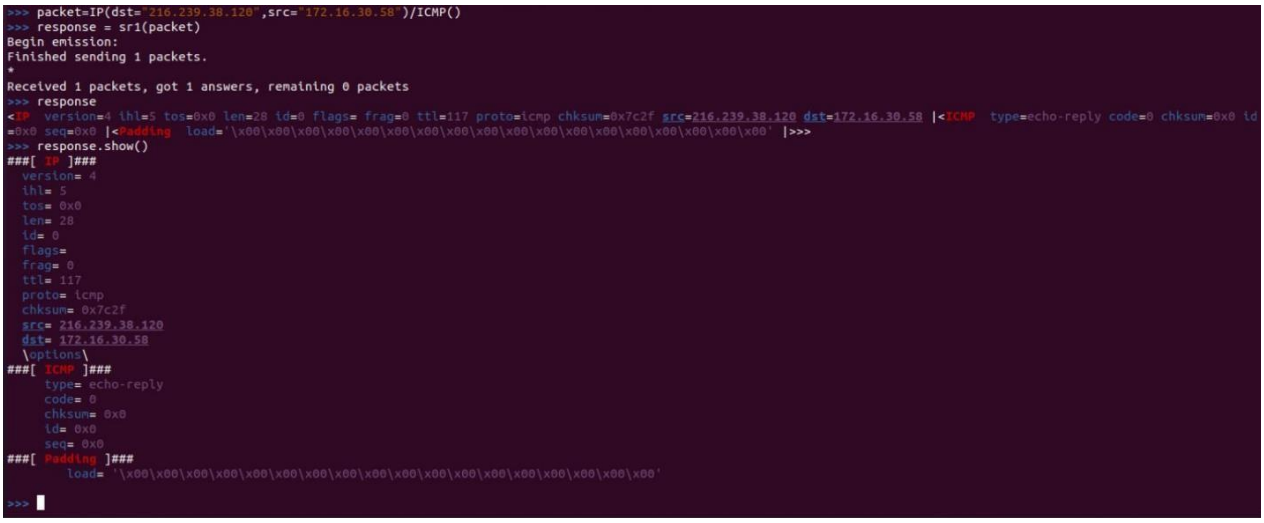
BHARATIYA VIDYA BHAVAN’S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
(Empowered Autonomous Institute Affiliated to Mumbai University)
Department Of Computer Engineering

Name	Manish Shashikant Jadhav
UID	2023301005
Subject	Computer Communication and Networks (CCN)
Experiment No.	7
Aim	Packet Crafting using Scapy

Step1: Ping (ICMP Echo Request):

- **Craft an ICMP Echo Request packet using Scapy.**
- **Send the packet to a target IP address.**
- **Expect an ICMP Echo Reply packet in response from the target.**

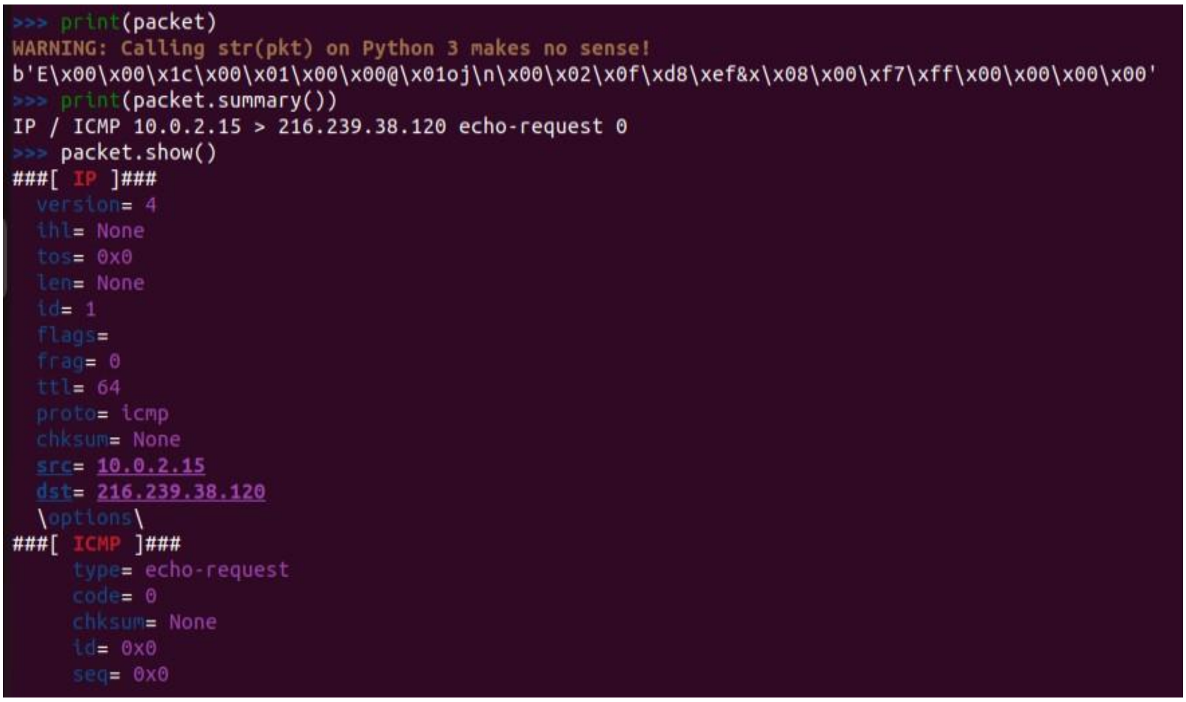
Crafting the packet and the response packet received:



```
>>> packet=IP(dst="216.239.38.120",src="172.16.30.58")/ICMP()
>>> response = sr1(packet)
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
>>> response
<IP  version=4 ihl=5 tos=0x0 len=28 id=0 flags= frag=0 ttl=117 proto=icmp chksum=0x7c2f src=216.239.38.120 dst=172.16.30.58 |<ICMP  type=echo-reply code=0 chksum=0x0 id
=0x0 seq=0x0 |<Padding  load='\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00' |>>>
>>> response.show()
###[ IP ]###
  version= 4
  ihl= 5
  tos= 0x0
  len= 28
  id= 0
  flags=
  frag= 0
  ttl= 117
  proto= icmp
  chksum= 0x7c2f
  src= 216.239.38.120
  dst= 172.16.30.58
  \options\
###[ ICMP ]###
     type= echo-reply
     code= 0
     chksum= 0x0
     id= 0x0
     seq= 0x0
###[ Padding ]###
        load= '\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'

>>> 
```

Sent Packet:

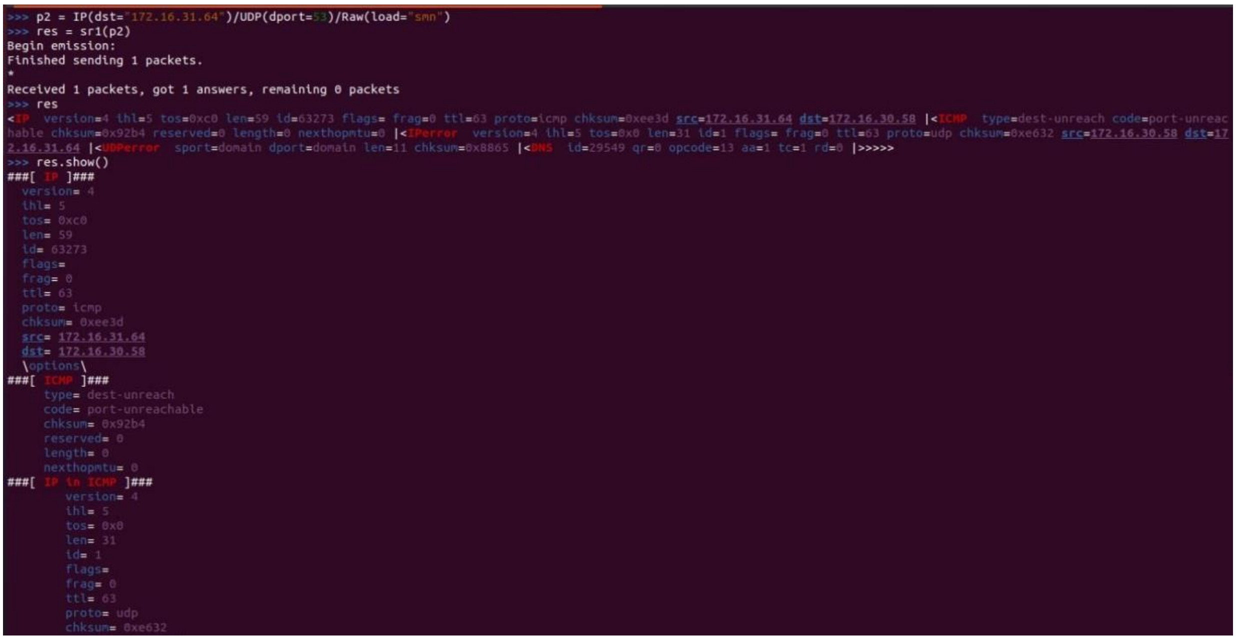


```
>>> print(packet)
WARNING: Calling str(pkt) on Python 3 makes no sense!
b'E\x00\x00\x1c\x00\x01\x00\x00@\x01oj\n\x00\x02\x0f\xd8\xef&x\x08\x00\xf7\xff\x00\x00\x00\x00'
>>> print(packet.summary())
IP / ICMP 10.0.2.15 > 216.239.38.120 echo-request 0
>>> packet.show()
###[ IP ]###
  version= 4
  ihl= None
  tos= 0x0
  len= None
  id= 1
  flags=
  frag= 0
  ttl= 64
  proto= icmp
  chksum= None
  src= 10.0.2.15
  dst= 216.239.38.120
  \options\
###[ ICMP ]###
     type= echo-request
     code= 0
     chksum= None
     id= 0x0
     seq= 0x0
```

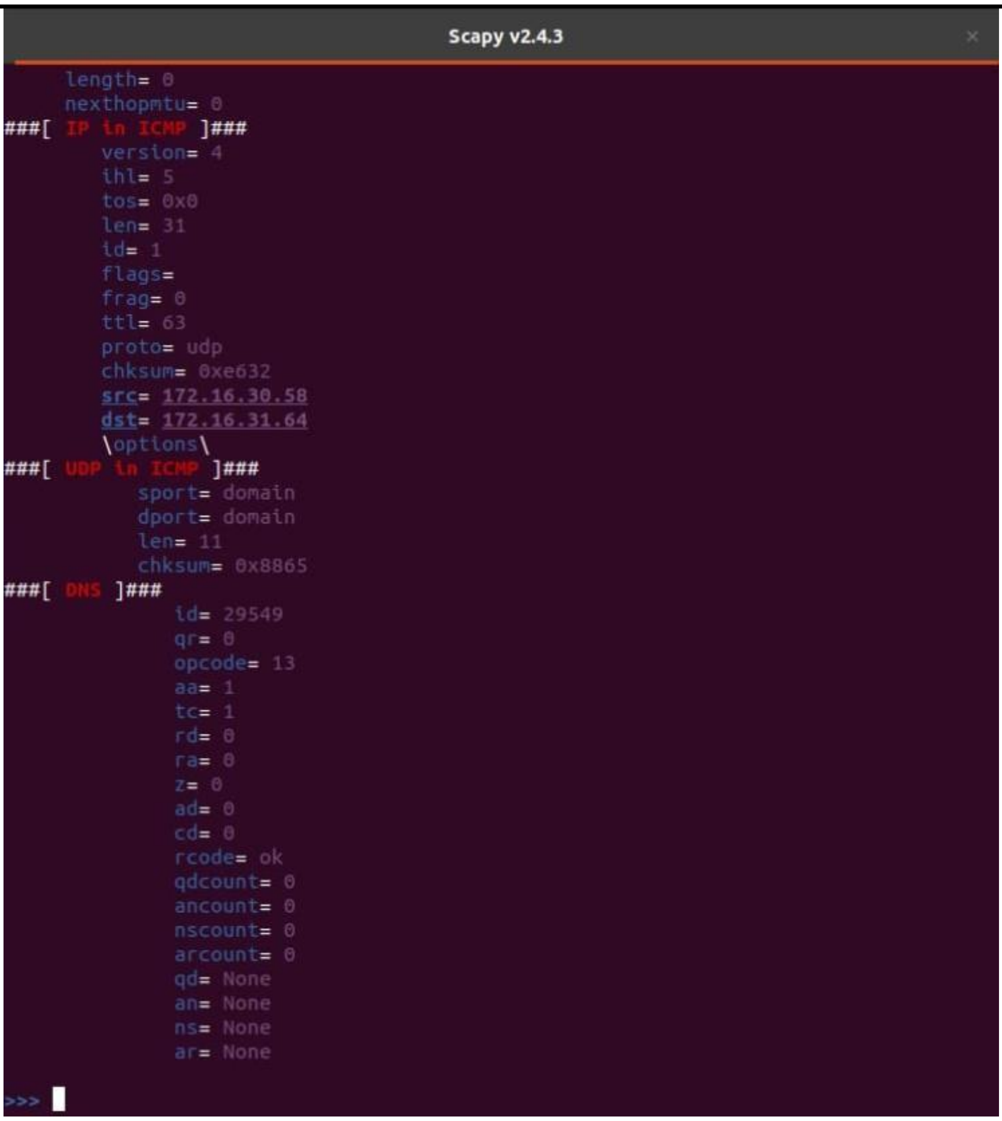
Step2: UDP Datagram

- **Craft a UDP packet with custom payload using Scapy.**
- **Send the UDP packet to a target listening on a specific UDP port.**
- **Expect a response from the target if the port is open and reachable.**

Crafting the packet and the response packet received:



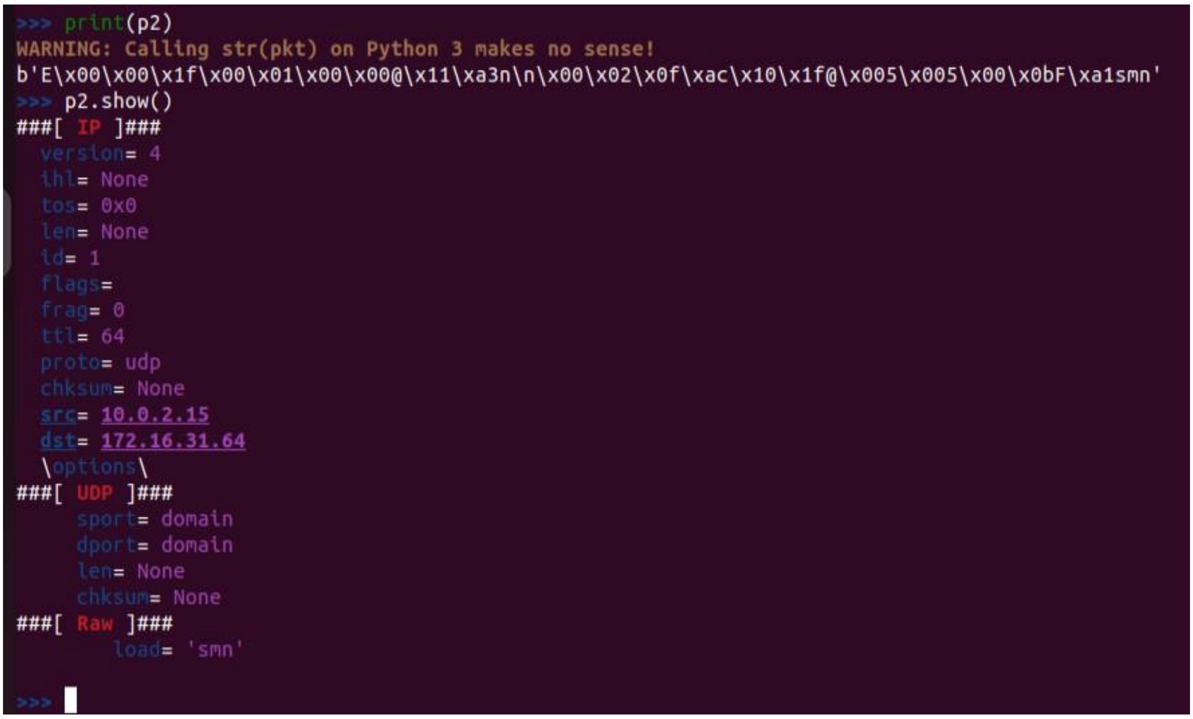
```
>>> p2 = IP(dst="172.16.31.64")/UDP(dport=53)/Raw(load="smn")
>>> res = sr1(p2)
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
>>> res
<IP  version=4 ihl=5 tos=0xc0 len=59 id=63273 flags= frag=0 ttl=63 proto=icmp chksum=0xee3d src=172.16.31.64 dst=172.16.30.58 |<ICMP  type=dest-unreach code=port-unreac
hable chksum=0x92b4 reserved=0 length=0 nexthopmtu=0 |<IPerror  version=4 ihl=5 tos=0x0 len=31 id=1 flags= frag=0 ttl=63 proto=udp chksum=0xe632 src=172.16.30.58 dst=17
2.16.31.64 |<UDPerror  sport=domain dport=domain len=11 chksum=0x8865 |<DNS  id=29549 qr=0 opcode=13 aa=1 tc=1 rd=0 |>>>>>
>>> res.show()
###[ IP ]###
  version= 4
  ihl= 5
  tos= 0xc0
  len= 59
  id= 63273
  flags=
  frag= 0
  ttl= 63
  proto= icmp
  chksum= 0xee3d
  src= 172.16.31.64
  dst= 172.16.30.58
  \options\
###[ ICMP ]###
     type= dest-unreach
     code= port-unreachable
     chksum= 0x92b4
     reserved= 0
     length= 0
     nexthopmtu= 0
###[ IP in ICMP ]###
        version= 4
        ihl= 5
        tos= 0x0
        len= 31
        id= 1
        flags=
        frag= 0
        ttl= 63
        proto= udp
        chksum= 0xe632
```



```
     length= 0
     nexthopmtu= 0
###[ IP in ICMP ]###
        version= 4
        ihl= 5
        tos= 0x0
        len= 31
        id= 1
        flags=
        frag= 0
        ttl= 63
        proto= udp
        chksum= 0xe632
        src= 172.16.30.58
        dst= 172.16.31.64
        \options\
###[ UDP in ICMP ]###
           sport= domain
           dport= domain
           len= 11
           chksum= 0x8865
###[ DNS ]###
              id= 29549
              qr= 0
              opcode= 13
              aa= 1
              tc= 1
              rd= 0
              ra= 0
              z= 0
              ad= 0
              cd= 0
              rcode= ok
              qdcount= 0
              ancount= 0
              nscount= 0
              arcount= 0
              qd= None
              an= None
              ns= None
              ar= None

>>> 
```

Sent Packet:



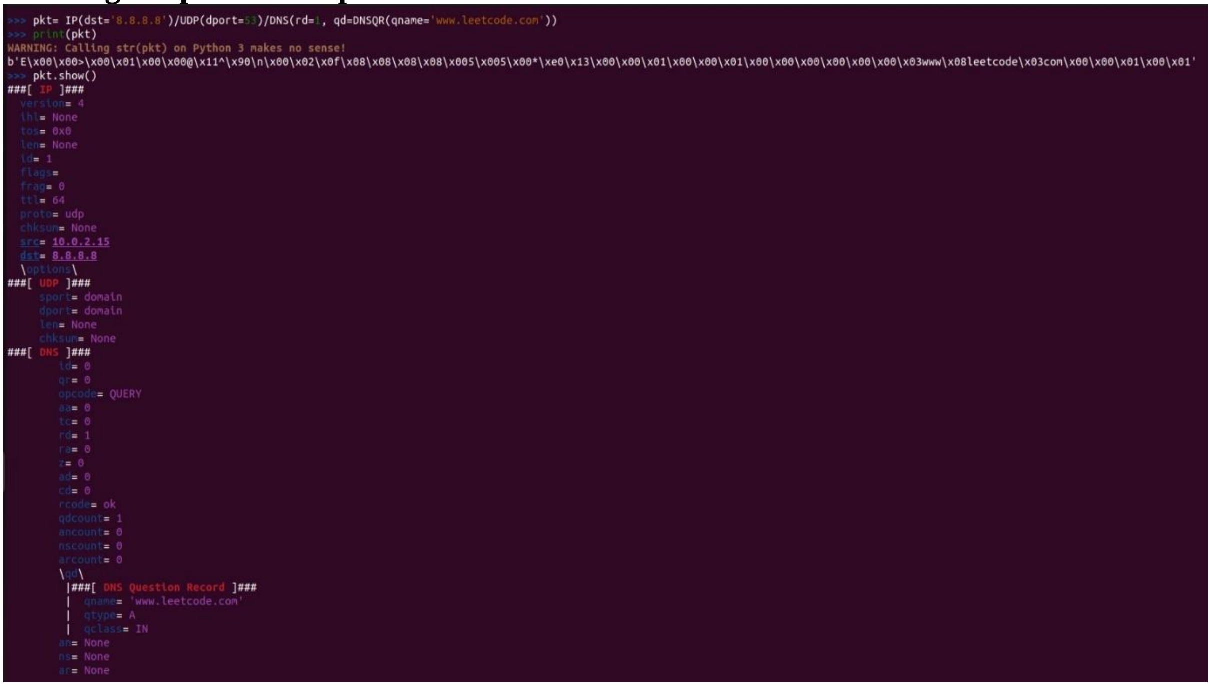
```
>>> print(p2)
WARNING: Calling str(pkt) on Python 3 makes no sense!
b'E\x00\x00\x1f\x00\x01\x00\x00@\x11\xa3n\n\x00\x02\x0f\xac\x10\x1f@\x005\x005\x00\x0bF\xa1smn'
>>> p2.show()
###[ IP ]###
  version= 4
  ihl= None
  tos= 0x0
  len= None
  id= 1
  flags=
  frag= 0
  ttl= 64
  proto= udp
  chksum= None
  src= 10.0.2.15
  dst= 172.16.31.64
  \options\
###[ UDP ]###
     sport= domain
     dport= domain
     len= None
     chksum= None
###[ Raw ]###
        load= 'smn'

>>> 
```

Step 3: DNS Query

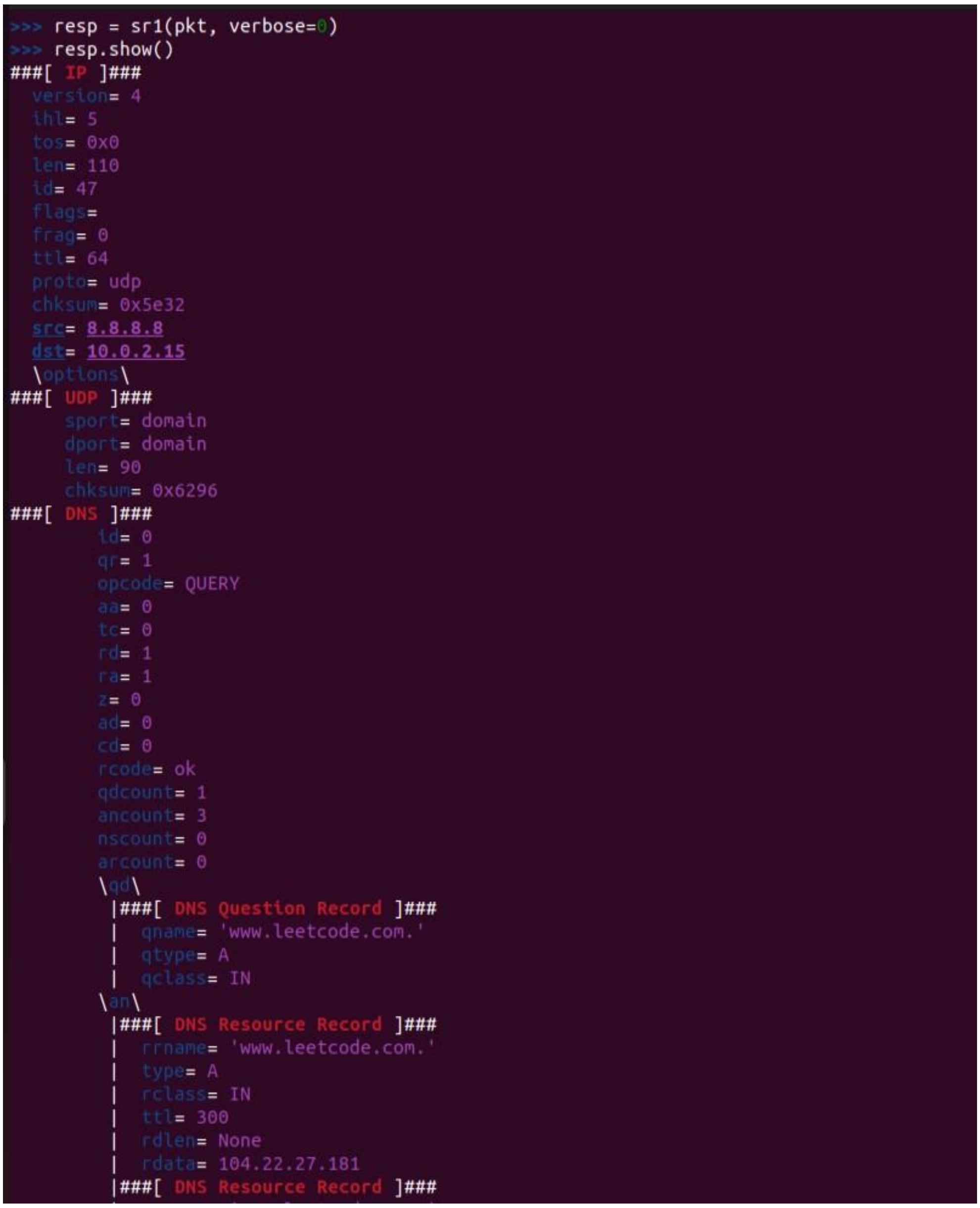
- **Craft a DNS query packet using Scapy to query a DNS server for a specific domain.**
- **Send the DNS query packet to the DNS server.**
- **Expect a DNS response containing the IP address associated with the queried domain.**

Crafting the packet and packet which is sent:

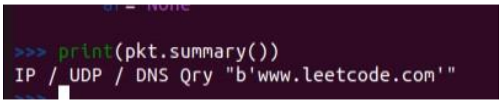


```
>>> pkt= IP(dst='8.8.8.8')/UDP(dport=53)/DNS(rd=1, qd=DNSQR(qname='www.leetcode.com'))
>>> print(pkt)
WARNING: Calling str(pkt) on Python 3 makes no sense!
b'E\x00\x00>\x00\x01\x00\x00@\x11^\x90\n\x00\x02\x0f\x08\x08\x08\x08\x005\x005\x00*\xe0\x13\x00\x00\x01\x00\x00\x01\x00\x00\x00\x00\x00\x00\x03www\x08leetcode\x03com\x00\x00\x01\x00\x01'
>>> pkt.show()
###[ IP ]###
  version= 4
  ihl= None
  tos= 0x0
  len= None
  id= 1
  flags=
  frag= 0
  ttl= 64
  proto= udp
  chksum= None
  src= 10.0.2.15
  dst= 8.8.8.8
  \options\
###[ UDP ]###
     sport= domain
     dport= domain
     len= None
     chksum= None
###[ DNS ]###
        id= 0
        qr= 0
        opcode= QUERY
        aa= 0
        tc= 0
        rd= 1
        ra= 0
        z= 0
        ad= 0
        cd= 0
        rcode= ok
        qdcount= 1
        ancount= 0
        nscount= 0
        arcount= 0
        \qd\
         |###[ DNS Question Record ]###
         |  qname= 'www.leetcode.com'
         |  qtype= A
         |  qclass= IN
        an= None
        ns= None
        ar= None
```

Response Packet:



```
>>> resp = sr1(pkt, verbose=0)
>>> resp.show()
###[ IP ]###
  version= 4
  ihl= 5
  tos= 0x0
  len= 110
  id= 47
  flags=
  frag= 0
  ttl= 64
  proto= udp
  chksum= 0x5e32
  src= 8.8.8.8
  dst= 10.0.2.15
  \options\
###[ UDP ]###
     sport= domain
     dport= domain
     len= 90
     chksum= 0x6296
###[ DNS ]###
        id= 0
        qr= 1
        opcode= QUERY
        aa= 0
        tc= 0
        rd= 1
        ra= 1
        z= 0
        ad= 0
        cd= 0
        rcode= ok
        qdcount= 1
        ancount= 3
        nscount= 0
        arcount= 0
        \qd\
         |###[ DNS Question Record ]###
         |  qname= 'www.leetcode.com.'
         |  qtype= A
         |  qclass= IN
        \an\
         |###[ DNS Resource Record ]###
         |  rrname= 'www.leetcode.com.'
         |  type= A
         |  rclass= IN
         |  ttl= 300
         |  rdlen= None
         |  rdata= 104.22.27.181
         |###[ DNS Resource Record ]###
```



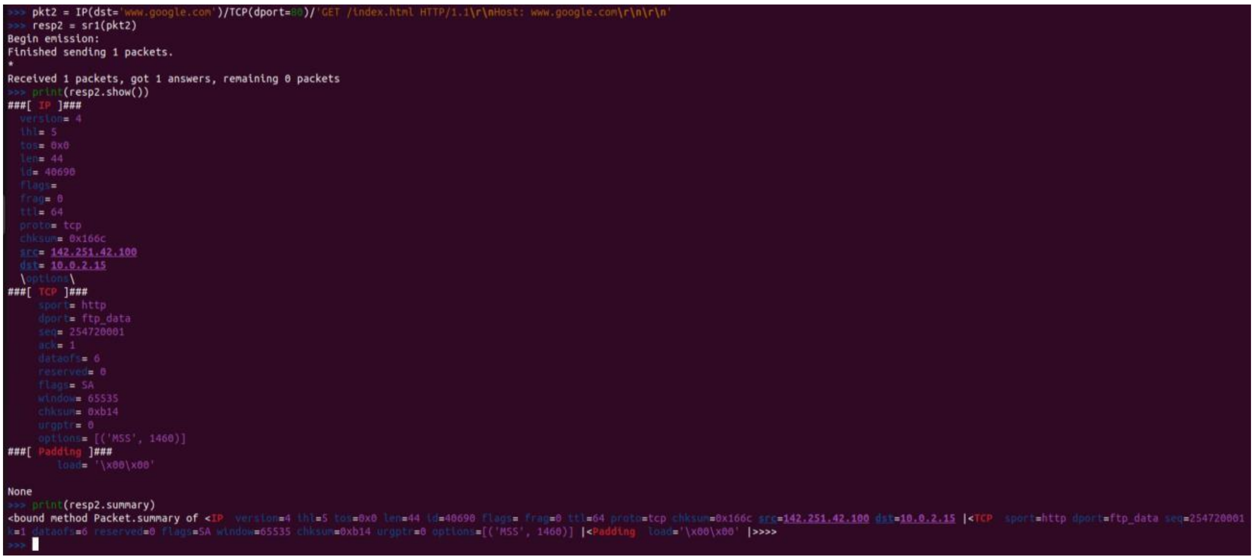
```
        ar= None

>>> print(pkt.summary())
IP / UDP / DNS Qry "b'www.leetcode.com'"
>>> 
```

Step4: HTTP GET Request

- **Craft an HTTP GET request packet using Scapy to retrieve a specific web page from a web server.**
- **Send the HTTP GET request to the web server.**
- **Expect an HTTP response containing the requested web page content.**

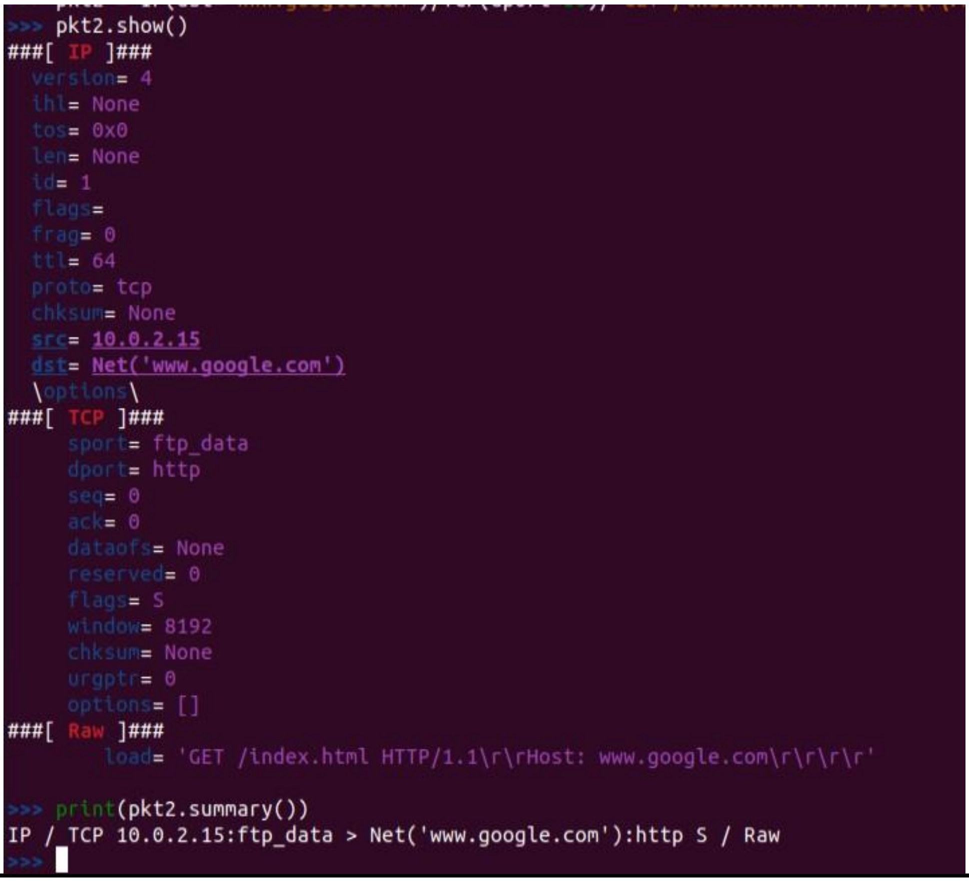
Crafting packets and response packet:



```
>>> pkt2 = IP(dst='www.google.com')/TCP(dport=80)/'GET /index.html HTTP/1.1\r\nHost: www.google.com\r\n\r\n'
>>> resp2 = sr1(pkt2)
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
>>> print(resp2.show())
###[ IP ]###
  version= 4
  ihl= 5
  tos= 0x0
  len= 44
  id= 40690
  flags=
  frag= 0
  ttl= 64
  proto= tcp
  chksum= 0x166c
  src= 142.251.42.100
  dst= 10.0.2.15
  \options\
###[ TCP ]###
     sport= http
     dport= ftp_data
     seq= 254720001
     ack= 1
     dataofs= 6
     reserved= 0
     flags= SA
     window= 65535
     chksum= 0xb14
     urgptr= 0
     options= [('MSS', 1460)]
###[ Padding ]###
        load= '\x00\x00'

None
>>> print(resp2.summary)
<bound method Packet.summary of <IP  version=4 ihl=5 tos=0x0 len=44 id=40690 flags= frag=0 ttl=64 proto=tcp chksum=0x166c src=142.251.42.100 dst=10.0.2.15 |<TCP  sport=http dport=ftp_data seq=254720001
k=1 dataofs=6 reserved=0 flags=SA window=65535 chksum=0xb14 urgptr=0 options=[('MSS', 1460)] |<Padding  load='\x00\x00' |>>>>
>>> 
```

Sent Packet:

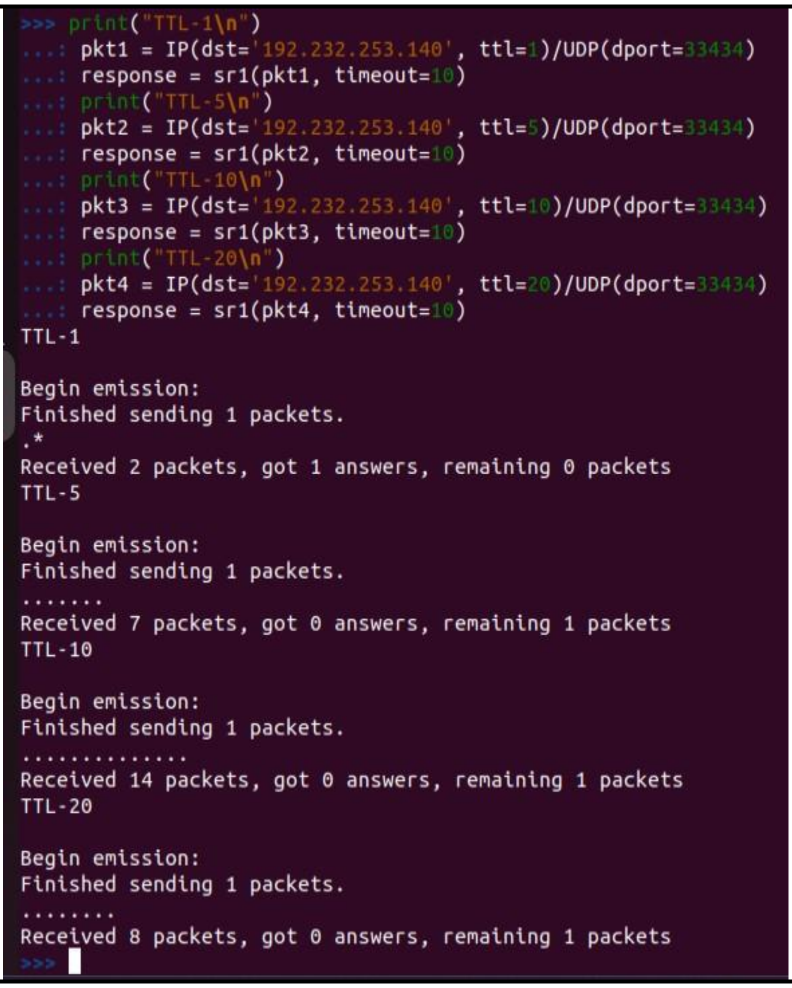


```
>>> pkt2.show()
###[ IP ]###
  version= 4
  ihl= None
  tos= 0x0
  len= None
  id= 1
  flags=
  frag= 0
  ttl= 64
  proto= tcp
  chksum= None
  src= 10.0.2.15
  dst= Net('www.google.com')
  \options\
###[ TCP ]###
     sport= ftp_data
     dport= http
     seq= 0
     ack= 0
     dataofs= None
     reserved= 0
     flags= S
     window= 8192
     chksum= None
     urgptr= 0
     options= []
###[ Raw ]###
        load= 'GET /index.html HTTP/1.1\r\rHost: www.google.com\r\r\r\r'

>>> print(pkt2.summary())
IP / TCP 10.0.2.15:ftp_data > Net('www.google.com'):http S / Raw
>>> 
```

Step6: Traceroute

- **Craft UDP packets with increasing TTL (Time-to-Live) values using Scapy.**
- **Send these packets towards a destination IP address.**
- **Observe the ICMP Time Exceeded messages returned by intermediate routers to map thenetwork path to the destination.**



```
>>> print("TTL-1\n")
...: pkt1 = IP(dst='192.232.253.140', ttl=1)/UDP(dport=33434)
...: response = sr1(pkt1, timeout=10)
...: print("TTL-5\n")
...: pkt2 = IP(dst='192.232.253.140', ttl=5)/UDP(dport=33434)
...: response = sr1(pkt2, timeout=10)
...: print("TTL-10\n")
...: pkt3 = IP(dst='192.232.253.140', ttl=10)/UDP(dport=33434)
...: response = sr1(pkt3, timeout=10)
...: print("TTL-20\n")
...: pkt4 = IP(dst='192.232.253.140', ttl=20)/UDP(dport=33434)
...: response = sr1(pkt4, timeout=10)
TTL-1

Begin emission:
Finished sending 1 packets.
.*
Received 2 packets, got 1 answers, remaining 0 packets
TTL-5

Begin emission:
Finished sending 1 packets.
.......
Received 7 packets, got 0 answers, remaining 1 packets
TTL-10

Begin emission:
Finished sending 1 packets.
..............
Received 14 packets, got 0 answers, remaining 1 packets
TTL-20

Begin emission:
Finished sending 1 packets.
........
Received 8 packets, got 0 answers, remaining 1 packets
>>> 
```

Conclusion	Hence, by completing this experiment I came to know about Packet Crafting using Scapy