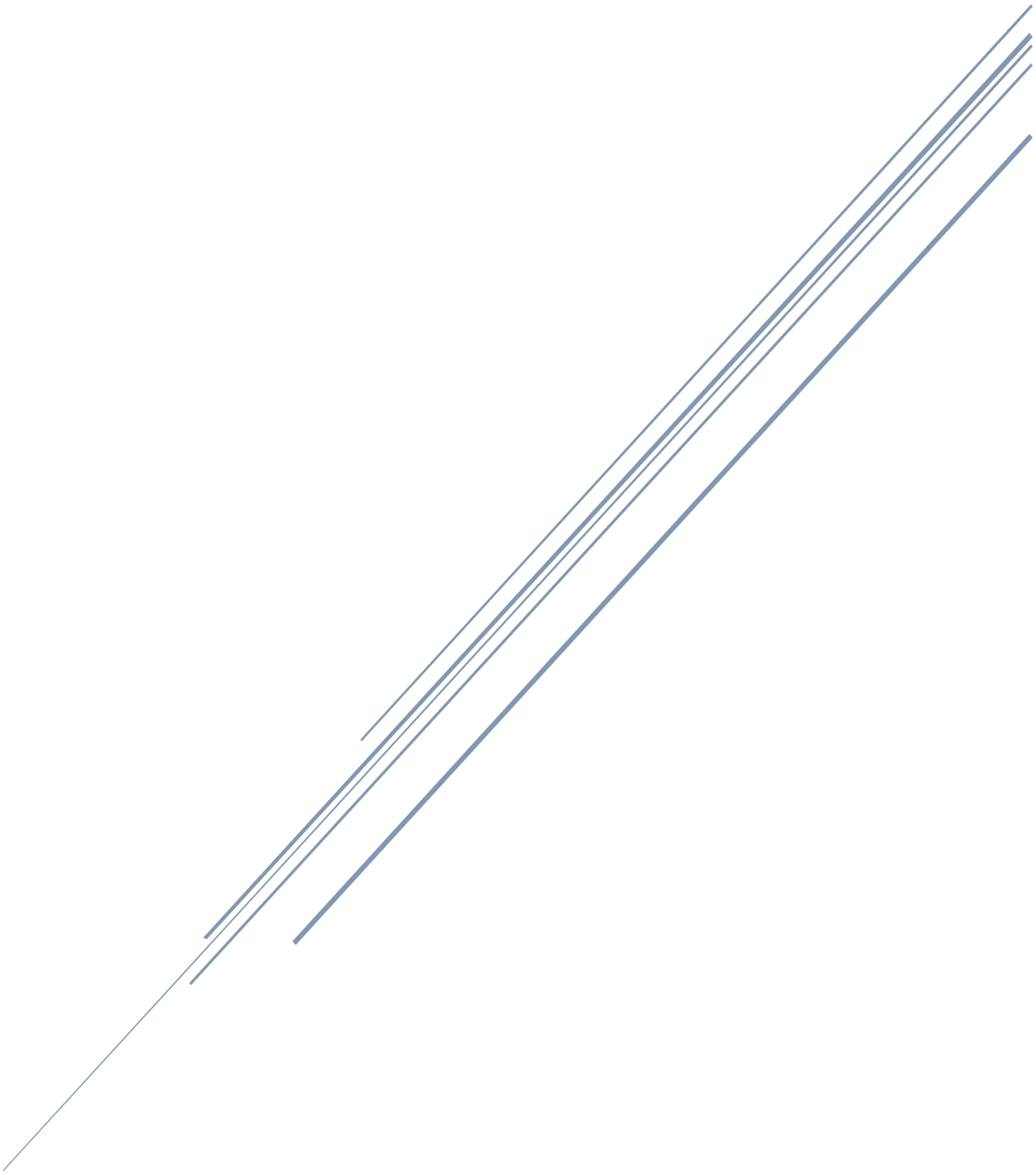
INTERNET SECURITY LAB 7

VPN TASK 1 and 2



Akarsh Shetty Umesh Mudelkadi
317752264

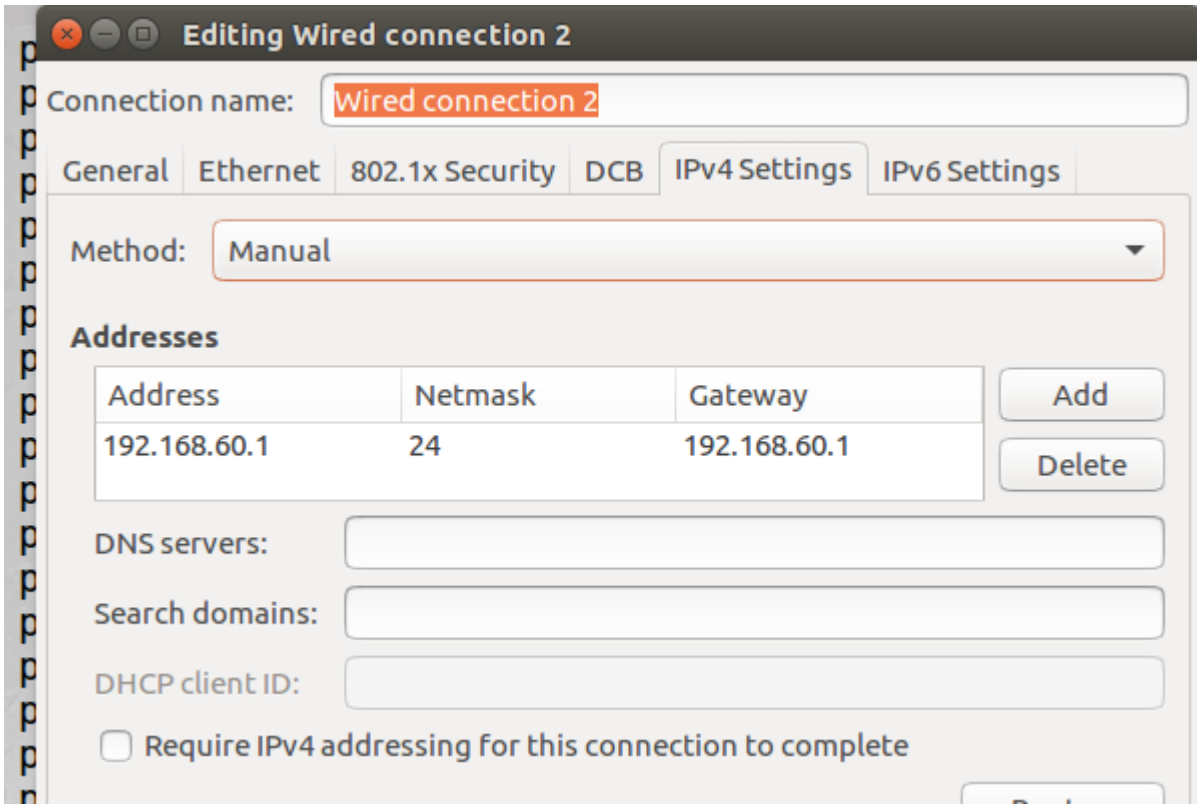
VM1 – Client

VM2 – Server

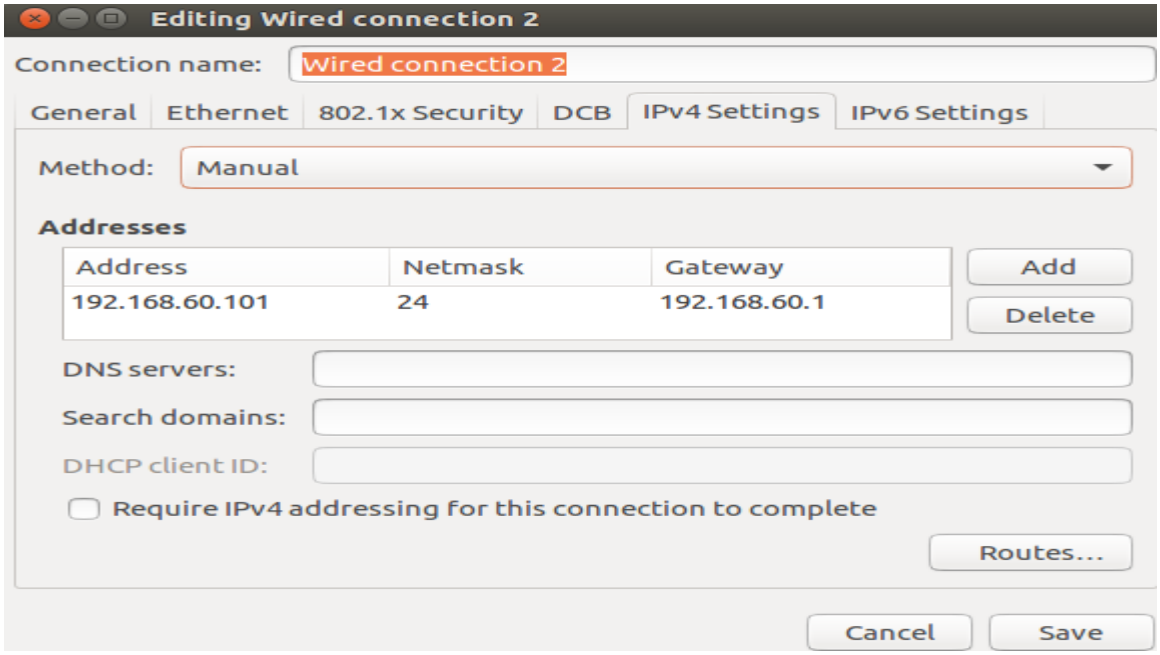
VM3 – Host V

Task 1:

Server side set up



Host V set up:



Task 2:

- **Step 1 – Started server program and initialized the IP for tun 0 interface and activated it, by command: sudo ifconfig tun0 192.168.53.1/24 up**
  Also enabled forwarding by command : sudo sysctl net.ipv4.ip_forward=1

```
tun0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          inet addr:192.168.53.5  P-t-P:192.168.53.5  Mask:255.255.255.0
          inet6 addr: fe80::7647:7f8a:748f:ce2e/64 Scope:Link
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:48 (48.0 B)

[03/29/2019 17:50]Mudelkadi@VM1:~$
```

Server program:

```
#include <fcntl.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <arpa/inet.h>
#include <linux/if.h>
#include <linux/if_tun.h>
#include <sys/ioctl.h>
```

```
#define PORT_NUMBER 55555
#define BUFF_SIZE 2000

struct sockaddr_in peerAddr;

int createTunDevice() {
  int tunfd;
  struct ifreq ifr;
  memset(&ifr, 0, sizeof(ifr));

  ifr.ifr_flags = IFF_TUN | IFF_NO_PI;

  tunfd = open("/dev/net/tun", O_RDWR);
  ioctl(tunfd, TUNSETIFF, &ifr);

  return tunfd;
}

int initUDPServer() {
   int sockfd;
   struct sockaddr_in server;
   char buff[100];

   memset(&server, 0, sizeof(server));
   server.sin_family = AF_INET;
   server.sin_addr.s_addr = htonl(INADDR_ANY);
   server.sin_port = htons(PORT_NUMBER);

   sockfd = socket(AF_INET, SOCK_DGRAM, 0);
   bind(sockfd, (struct sockaddr*) &server, sizeof(server));

   // Wait for the VPN client to "connect".
   bzero(buff, 100);
   int peerAddrLen = sizeof(struct sockaddr_in);
   int len = recvfrom(sockfd, buff, 100, 0,
            (struct sockaddr *) &peerAddr, &peerAddrLen);

   printf("Connected with the client: %s\n", buff);
   return sockfd;
}

void tunSelected(int tunfd, int sockfd){
   int  len;
   char buff[BUFF_SIZE];

   printf("Got a packet from TUN\n");
```

```
    bzero(buff, BUFF_SIZE);
    len = read(tunfd, buff, BUFF_SIZE);
    sendto(sockfd, buff, len, 0, (struct sockaddr *) &peerAddr,
                    sizeof(peerAddr));
}

void socketSelected (int tunfd, int sockfd){
    int  len;
    char buff[BUFF_SIZE];

    printf("Got a packet from the tunnel\n");

    bzero(buff, BUFF_SIZE);
    len = recvfrom(sockfd, buff, BUFF_SIZE, 0, NULL, NULL);
    write(tunfd, buff, len);

}
int main (int argc, char * argv[]) {
   int tunfd, sockfd;

   tunfd  = createTunDevice();
   sockfd = initUDPServer();

   // Enter the main loop
   while (1) {
     fd_set readFDSet;

     FD_ZERO(&readFDSet);
     FD_SET(sockfd, &readFDSet);
     FD_SET(tunfd, &readFDSet);
     select(FD_SETSIZE, &readFDSet, NULL, NULL, NULL);

     if (FD_ISSET(tunfd,  &readFDSet)) tunSelected(tunfd, sockfd);
     if (FD_ISSET(sockfd, &readFDSet)) socketSelected(tunfd, sockfd);
   }
}
```

- **Step 2: Started Client program and initialized the IP for tun 0 interface and activated it, by command sudo ifconfig tun0 192.168.53.5/24 up.**

```
tun0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          inet addr:192.168.53.5  P-t-P:192.168.53.5  Mask:255.255.255.0
          inet6 addr: fe80::7647:7f8a:748f:ce2e/64 Scope:Link
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:48 (48.0 B)
```

Client program:

```
#include <fcntl.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <arpa/inet.h>
#include <linux/if.h>
#include <linux/if_tun.h>
#include <sys/ioctl.h>

#define BUFF_SIZE 2000
#define PORT_NUMBER 55555
#define SERVER_IP "10.0.2.4"
struct sockaddr_in peerAddr;

int createTunDevice() {
  int tunfd;
  struct ifreq ifr;
  memset(&ifr, 0, sizeof(ifr));

  ifr.ifr_flags = IFF_TUN | IFF_NO_PI;

  tunfd = open("/dev/net/tun", O_RDWR);
  ioctl(tunfd, TUNSETIFF, &ifr);

  return tunfd;
}

int connectToUDPServer(){
   int sockfd;
   char *hello="Hello";

   memset(&peerAddr, 0, sizeof(peerAddr));
   peerAddr.sin_family = AF_INET;
   peerAddr.sin_port = htons(PORT_NUMBER);
   peerAddr.sin_addr.s_addr = inet_addr(SERVER_IP);
```

```
    sockfd = socket(AF_INET, SOCK_DGRAM, 0);

    // Send a hello message to "connect" with the VPN server
    sendto(sockfd, hello, strlen(hello), 0,
            (struct sockaddr *) &peerAddr, sizeof(peerAddr));

    return sockfd;
}



void tunSelected(int tunfd, int sockfd){
    int  len;
    char buff[BUFF_SIZE];

    printf("Got a packet from TUN\n");

    bzero(buff, BUFF_SIZE);
    len = read(tunfd, buff, BUFF_SIZE);
    sendto(sockfd, buff, len, 0, (struct sockaddr *) &peerAddr,
                sizeof(peerAddr));
}

void socketSelected (int tunfd, int sockfd){
    int  len;
    char buff[BUFF_SIZE];

    printf("Got a packet from the tunnel\n");

    bzero(buff, BUFF_SIZE);
    len = recvfrom(sockfd, buff, BUFF_SIZE, 0, NULL, NULL);
    write(tunfd, buff, len);

}
int main (int argc, char * argv[]) {
   int tunfd, sockfd;

   tunfd  = createTunDevice();
   sockfd = connectToUDPServer();

   // Enter the main loop
   while (1) {
     fd_set readFDSet;

     FD_ZERO(&readFDSet);
     FD_SET(sockfd, &readFDSet);
     FD_SET(tunfd, &readFDSet);
```

```
    select(FD_SETSIZE, &readFDSet, NULL, NULL, NULL);

    if (FD_ISSET(tunfd,  &readFDSet)) tunSelected(tunfd, sockfd);
    if (FD_ISSET(sockfd, &readFDSet)) socketSelected(tunfd, sockfd);
  }
}
```

- **Step 3 and Step 4 : Routing tables for client, server and host v :**

  Client:

```
[03/29/2019 23:32]Mudelkadi@VM1:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref
  Use Iface
default         10.0.2.1        0.0.0.0         UG    100    0
    0 enp0s3
10.0.2.0        *               255.255.255.0   U     100    0
    0 enp0s3
link-local      *               255.255.0.0     U     1000   0
    0 enp0s3
192.168.53.0    *               255.255.255.0   U     0      0
    0 tun0
192.168.60.0    *               255.255.255.0   U     0      0
    0 tun0
```

Server:

```
[03/29/2019 23:34]Mudelkadi@VM2:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.60.1    0.0.0.0         UG    100    0        0 enp0s8
default         10.0.2.1        0.0.0.0         UG    101    0        0 enp0s3
10.0.2.0        *               255.255.255.0   U     100    0        0 enp0s3
link-local      *               255.255.0.0     U     1000   0        0 enp0s8
192.168.53.0    *               255.255.255.0   U     0      0        0 tun0
192.168.60.0    *               255.255.255.0   U     100    0        0 enp0s8
[03/29/2019 23:35]Mudelkadi@VM2:~$
```

Host V:

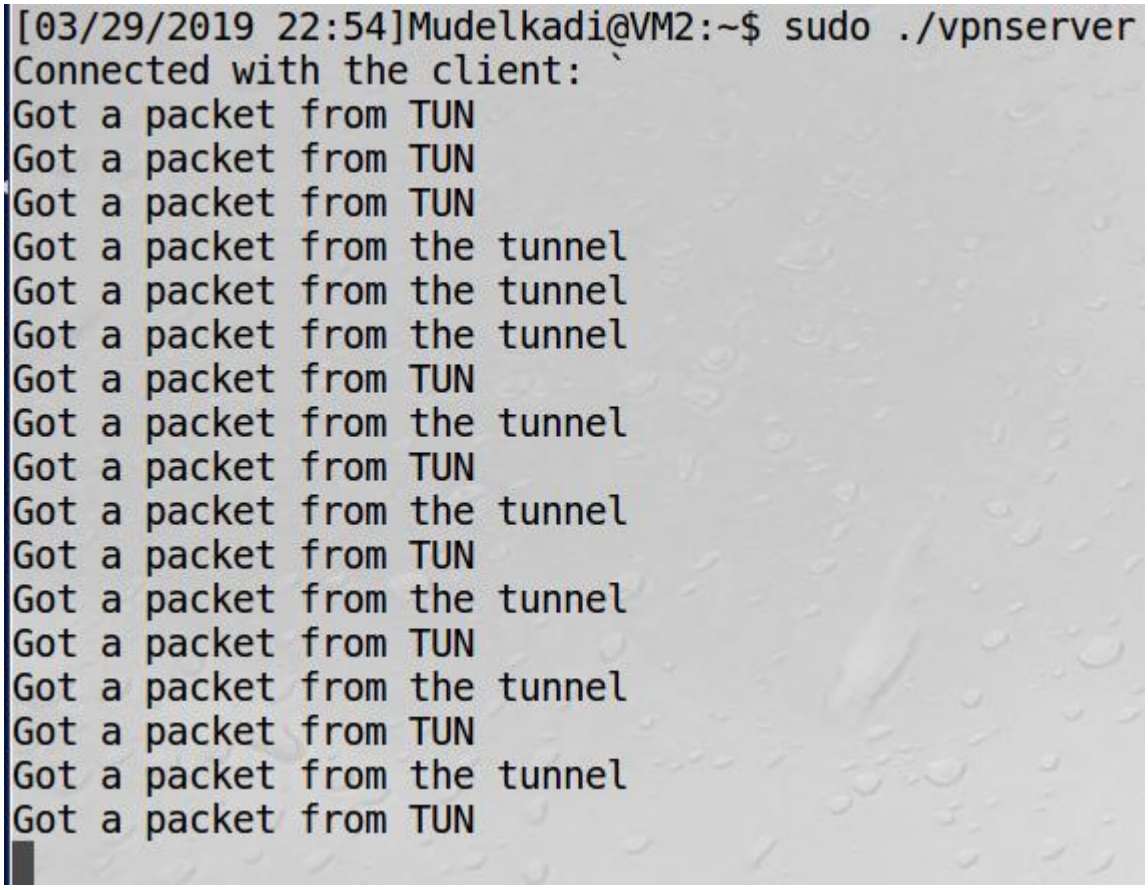
```
[03/29/2019 22:26]Mudelkadi@VM3:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.60.1    0.0.0.0         UG    100    0        0 enp0s8
link-local      *               255.255.0.0     U     1000   0        0 enp0s8
192.168.60.0    *               255.255.255.0   U     100    0        0 enp0s8
[03/29/2019 23:34]Mudelkadi@VM3:~$
```

- **Step 5: Testing the VPN table:**

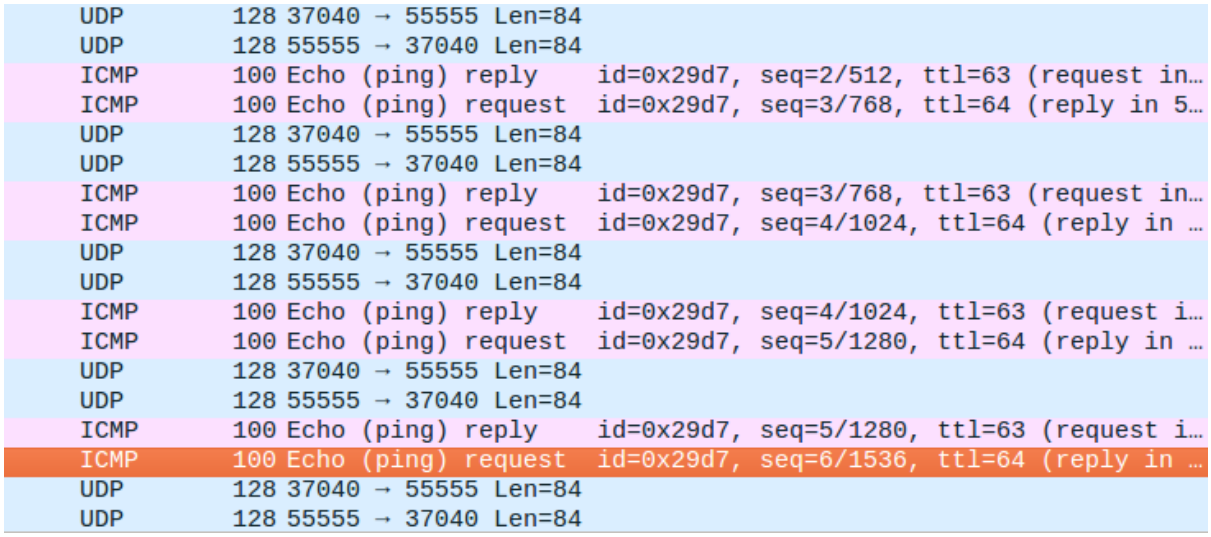
1) **Ping**

```
[03/29/2019 23:04]Mudelkadi@VM1:~$ ping 192.168.60.101
PING 192.168.60.101 (192.168.60.101) 56(84) bytes of data.
64 bytes from 192.168.60.101: icmp_seq=1 ttl=63 time=1.35 ms
64 bytes from 192.168.60.101: icmp_seq=2 ttl=63 time=1.13 ms
64 bytes from 192.168.60.101: icmp_seq=3 ttl=63 time=3.08 ms
64 bytes from 192.168.60.101: icmp_seq=4 ttl=63 time=2.90 ms
64 bytes from 192.168.60.101: icmp_seq=5 ttl=63 time=3.07 ms
64 bytes from 192.168.60.101: icmp_seq=6 ttl=63 time=3.16 ms
^C
--- 192.168.60.101 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
```

```
[03/29/2019 22:58]Mudelkadi@VM1:~$ sudo ./vpnclient
Got a packet from TUN
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from TUN
Got a packet from TUN
Got a packet from TUN
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
```



592	2019-03-29 23:04:13.4332469…	10.0.2.15	10.0.2.4	UI
593	2019-03-29 23:04:13.4342023…	10.0.2.4	10.0.2.15	UI
594	2019-03-29 23:04:13.4343002…	192.168.60.101	192.168.53.5	IC
595	2019-03-29 23:04:14.4352700…	192.168.53.5	192.168.60.101	IC
596	2019-03-29 23:04:14.4353940…	10.0.2.15	10.0.2.4	UI
597	2019-03-29 23:04:14.4380647…	10.0.2.4	10.0.2.15	UI
598	2019-03-29 23:04:14.4382917…	192.168.60.101	192.168.53.5	IC
599	2019-03-29 23:04:15.4367321…	192.168.53.5	192.168.60.101	IC
600	2019-03-29 23:04:15.4368530…	10.0.2.15	10.0.2.4	UI
601	2019-03-29 23:04:15.4393565…	10.0.2.4	10.0.2.15	UI
602	2019-03-29 23:04:15.4395750…	192.168.60.101	192.168.53.5	IC
603	2019-03-29 23:04:16.4378580…	192.168.53.5	192.168.60.101	IC
604	2019-03-29 23:04:16.4379757…	10.0.2.15	10.0.2.4	UI
605	2019-03-29 23:04:16.4406292…	10.0.2.4	10.0.2.15	UI
606	2019-03-29 23:04:16.4408711…	192.168.60.101	192.168.53.5	IC
607	2019-03-29 23:04:17.4399660…	192.168.53.5	192.168.60.101	IC
608	2019-03-29 23:04:17.4400838…	10.0.2.15	10.0.2.4	UI
609	2019-03-29 23:04:17.4428445…	10.0.2.4	10.0.2.15	UI



Observation – Wireshark shows that the packets across the tunnel are enclosed between 10.0.2.15 and 10.0.2.4. The actual communication happens between host tun0 interface and host V which is routed by VPN server. Same can be seen for **Telnet.** Screenshots are below for telnet. Tun0 packet transfers shown in pink.

Telnet Screenshots below:

```
[03/29/2019 23:16]Mudelkadi@VM1:~$ telnet 192.168.60.101
Trying 192.168.60.101...
Connected to 192.168.60.101.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Last login: Mon Feb 18 22:26:02 EST 2019 from 10.0.2.4 on pts/1
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

3 packages can be updated.
0 updates are security updates.
```

81	2019-03-29	23:16:48.3569622…	192.168.53.5	192.168.60.101	T
82	2019-03-29	23:16:48.3569993…	10.0.2.15	10.0.2.4	U
83	2019-03-29	23:16:48.3581478…	10.0.2.4	10.0.2.15	U
84	2019-03-29	23:16:48.3582467…	192.168.60.101	192.168.53.5	T
85	2019-03-29	23:16:48.3582742…	192.168.53.5	192.168.60.101	T
86	2019-03-29	23:16:48.3582926…	10.0.2.15	10.0.2.4	U
87	2019-03-29	23:16:48.3588095…	192.168.53.5	192.168.60.101	T
88	2019-03-29	23:16:48.3588361…	10.0.2.15	10.0.2.4	U
89	2019-03-29	23:16:48.3597088…	10.0.2.4	10.0.2.15	U
90	2019-03-29	23:16:48.3597899…	192.168.60.101	192.168.53.5	T
91	2019-03-29	23:16:48.5572653…	10.0.2.4	10.0.2.15	U
92	2019-03-29	23:16:48.5575266…	192.168.60.101	192.168.53.5	T
93	2019-03-29	23:16:48.5575688…	192.168.53.5	192.168.60.101	T
94	2019-03-29	23:16:48.5576492…	10.0.2.15	10.0.2.4	U
95	2019-03-29	23:16:48.5595308…	10.0.2.4	10.0.2.15	U
96	2019-03-29	23:16:48.5596561…	192.168.60.101	192.168.53.5	T

TCP	76	35798 → 23 [SYN] Seq=2696240571 Win=29200 Len=0 MSS=1460 SACK…
UDP	104	56914 → 55555 Len=60
UDP	104	55555 → 56914 Len=60
TCP	76	23 → 35798 [SYN, ACK] Seq=3239947134 Ack=2696240572 Win=28960…
TCP	68	35798 → 23 [ACK] Seq=2696240572 Ack=3239947135 Win=29312 Len=…
UDP	96	56914 → 55555 Len=52
TELNET	95	Telnet Data ...
UDP	123	56914 → 55555 Len=79
UDP	96	55555 → 56914 Len=52
TCP	68	23 → 35798 [ACK] Seq=3239947135 Ack=2696240599 Win=29056 Len=…
UDP	108	55555 → 56914 Len=64
TELNET	80	Telnet Data ...
TCP	68	35798 → 23 [ACK] Seq=2696240599 Ack=3239947147 Win=29312 Len=…
UDP	96	56914 → 55555 Len=52
UDP	135	55555 → 56914 Len=91
TELNET	107	Telnet Data ...

Step 6:

After the tunnel is broken the TCP retransmits packets sent by both host client and host V. Hence, once the connection is re-established these retransmitted packets synchronize and

follows the regular transfer of packets. This creates an affect where all the letters we typed when the tunnel was broken suddenly pop up on screen when the connection is re-established.