



**Air University, Aerospace and Aviation Campus, Kamra**

**Department of Computer Science**

**BS. Cyber Security**

**NETWORK SECURITY LAB**

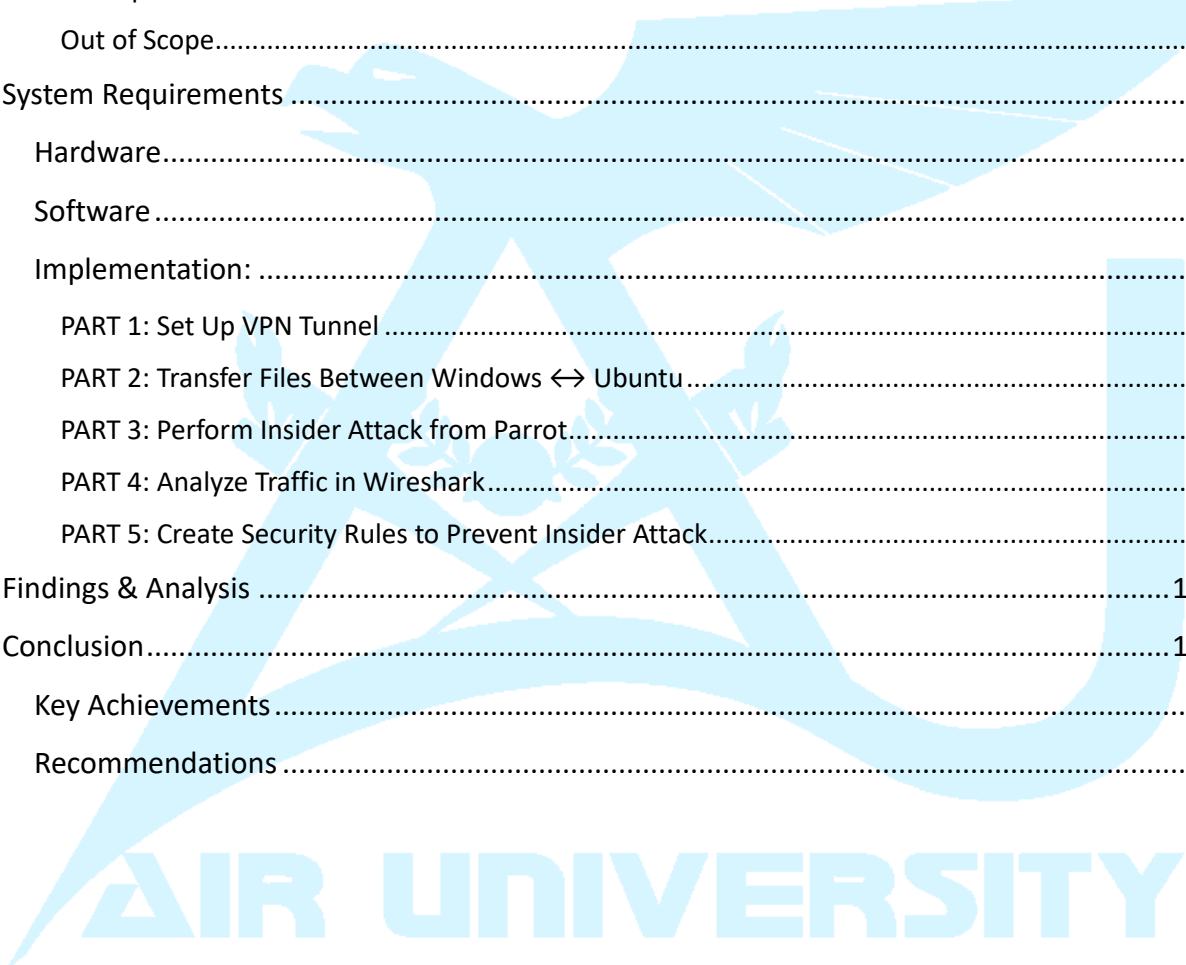
**SEMESTER LAB PROJECT REPORT**

**BSCYS – IV**

<b>Project Title</b>	Secure VPN Tunnel using OpenVPN
<b>Semester</b>	BS(CYS) – IV (Spring 2025)
<b>Number of group members</b>	Zunaira (235051) Javeria Gul (235031) Laiba (235043) Fakhra (235057) Maimona (235056)
<b>Date of Submission</b>	June 12,2025
<b>Submitted To</b>	<b>MAHNOOR GILLANI</b> Lecturer (Cyber Security) Department of Cyber Security, AACK
<b>Remarks by Teacher:</b>	

## Table of Contents

Project Overview .....	3
Objectives.....	3
Network Architecture: .....	3
IP Assignments: .....	3
Scope.....	4
In Scope.....	4
Out of Scope.....	4
System Requirements .....	5
Hardware.....	5
Software .....	5
Implementation: .....	5
PART 1: Set Up VPN Tunnel .....	5
PART 2: Transfer Files Between Windows ↔ Ubuntu.....	10
PART 3: Perform Insider Attack from Parrot.....	12
PART 4: Analyze Traffic in Wireshark.....	14
PART 5: Create Security Rules to Prevent Insider Attack.....	17
Findings & Analysis .....	18
Conclusion.....	19
Key Achievements .....	19
Recommendations .....	19



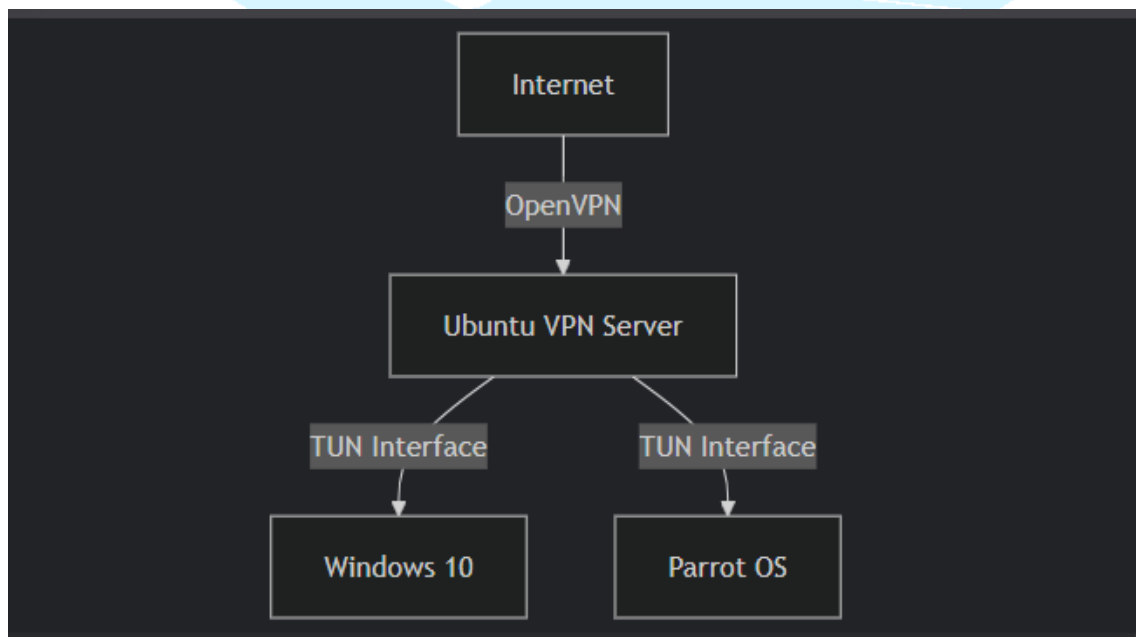
## Project Overview

### Objectives

This project demonstrates:

- Secure OpenVPN tunnel establishment between multiple OS platforms
- Encrypted file transfer verification
- Insider threat simulation and analysis
- Implementation of protective security measures

### Network Architecture:



### IP Assignments:

Virtual Machine	VPN Server IP	Lan Interface IP
Ubuntu Server	10.8.0.1	192.168.28.129
Windows Client	10.8.0.3	192.168.28.132
Parrot OS Client	10.8.0.2	192.168.28.133

## Scope

This project focuses on implementing and evaluating a secure VPN infrastructure with the following boundaries:

### In Scope

#### 1. VPN Implementation

- OpenVPN server setup on Ubuntu 20.04
- Client configurations for Windows 10 and Parrot OS
- Certificate-based authentication

#### 2. Security Testing

- Encrypted file transfer validation (SCP/SMB)
- Simulated insider attacks from compromised client
- Traffic encryption analysis using Wireshark

#### 3. Defensive Measures

- Firewall configuration (UFW/IPTables)
- Client-to-client access control
- Service hardening (SMB restrictions)

### Out of Scope

1. Enterprise-scale VPN deployment
2. Physical network infrastructure
3. Advanced Persistent Threat (APT) simulations
4. Mobile device connectivity

## System Requirements

### Hardware

Component	Specification
Server	Ubuntu 20.04 (2vCPU, 4GB RAM)
Client 1	Windows 10 (x64)
Client 2	Parrot OS (Security Edition)

### Software

- OpenVPN 2.4.7
- Easy-RSA 3.0.6
- Wireshark 3.4.0
- Nmap 7.80

### Implementation:

#### PART 1: Set Up VPN Tunnel

On Ubuntu (VPN Server)

- Update & install OpenVPN

```
zunaira@zunaira-VMware-Virtual-Platform:~$ sudo apt update
[sudo] password for zunaira:
Hit:1 http://pk.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://pk.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://pk.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21.6 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:7 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.1 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Fetched 200 kB in 4s (50.1 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
21 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

```
zunaira@zunaira-VMware-Virtual-Platform:~$ sudo apt install wget curl openvpn
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
wget is already the newest version (1.21.4-1ubuntu4.1).
curl is already the newest version (8.5.0-2ubuntu10.6).
openvpn is already the newest version (2.6.12-0ubuntu0.24.04.3).
0 upgraded, 0 newly installed, 0 to remove and 21 not upgraded.
```

Download & run install script:

```
zunaira@zunaira-VMware-Virtual-Platform:~$ wget https://git.io/vpn -O openvpn-install.sh
--2025-06-10 17:04:41-- https://git.io/vpn
Resolving git.io (git.io)... 140.82.113.21
Connecting to git.io (git.io)|140.82.113.21|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://raw.githubusercontent.com/Nyr/openvpn-install/master/openvpn-install.sh [following]
--2025-06-10 17:04:43-- https://raw.githubusercontent.com/Nyr/openvpn-install/master/openvpn-install.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://raw.githubusercontent.com/Nyr/openvpn-install/master/openvpn-install.sh [following]
--2025-06-10 17:04:44-- https://raw.githubusercontent.com/Nyr/openvpn-install/master/openvpn-install.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 24807 (24K) [text/plain]
Saving to: 'openvpn-install.sh'

openvpn-install.sh      100%[=====>] 24.23K  --.-KB/s   in 0.03s

2025-06-10 17:04:45 (807 KB/s) - 'openvpn-install.sh' saved [24807/24807]
```





Choose options during setup:

- Name: client1 (for Windows)

[illegible]

```

Notice
-----
Private-Key and Public-Certificate-Request files created.
Your files are:
* req: /etc/openvpn/server/easy-rsa/pki/reqs/client1.req
* key: /etc/openvpn/server/easy-rsa/pki/private/client1.key

Using configuration from /etc/openvpn/server/easy-rsa/pki/dfc148de/temp.6.1
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName             :ASN.1 12:'client1'
Certificate is to be certified until Jun  8 12:06:24 2035 GMT (3650 days)

Write out database with 1 new entries
Database updated

Notice
-----
Inline file created:
* /etc/openvpn/server/easy-rsa/pki/inline/private/client1.inline

```

Check for .ovpn file:

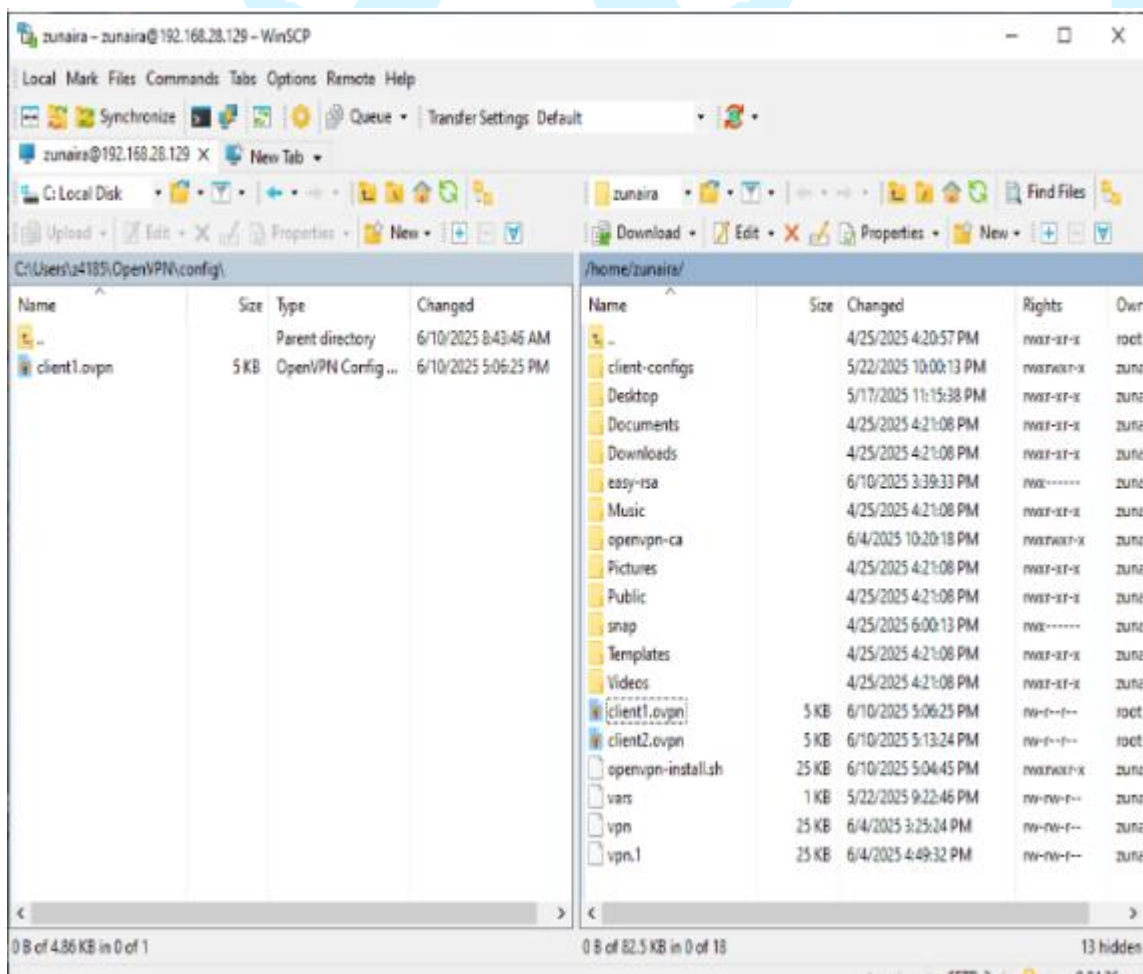
```
zunaira@zunaira-VMware-Virtual-Platform:~$ ls ~/.ovpn
/home/zunaira/client1.ovpn
```

On Windows 10 (Client)

- Install OpenVPN GUI



- Transfer .ovpn file (client1.ovpn) using WinSCP

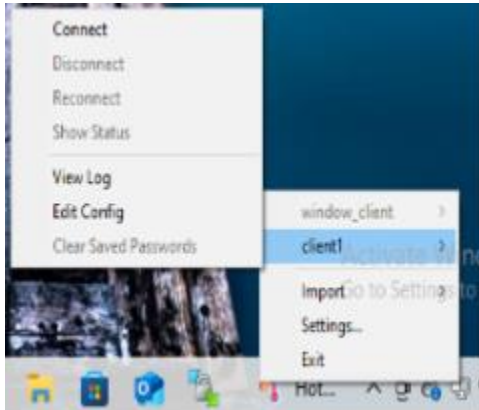




Place file in:

C:\Users\z4185\OpenVPN\config

- Right-click OpenVPN GUI → Run as Admin → Connect



On Parrot OS (Client/Attacker)

- Transfer client2.ovpn via SCP:

```
[x]-[user@parrot]-[~/Desktop]
$scp zunaira@192.168.28.129:/home/zunaira/client2.ovpn ~/
zunaira@192.168.28.129's password:
client2.ovpn 100% 4977 888.7KB/s 00:00
```

- Connect to VPN:

```
user@parrot:~$ sudo openvpn --config client2.ovpn
2025-06-10 17:29:30 --cipher is not set. Previous OpenVPN version defaulted to BF-CBC as fallback when cipher negotiation failed in this case. If you need this fallback please add '--data-ciphers-fallback BF-CBC' to your configuration and/or add BF-CBC to --data-ciphers.
2025-06-10 17:29:30 OpenVPN 2.5.1 x86_64-linux-gnu (OpenSSL) [LZO] [L24] [EPOLL] [PKCS11] [MH/PTWFO] [AEAD] built on Feb 24 2021
2025-06-10 17:29:30 library versions: OpenSSL 1.1.1k 25 Mar 2021, LZO 2.10
2025-06-10 17:29:30 Outgoing Control Channel Encryption: Cipher 'AES-256-CTR' initialized with 256 bit key
2025-06-10 17:29:30 Outgoing Control Channel Encryption: Using 256 bit message hash 'SHA256' for HMAC authentication
2025-06-10 17:29:30 Incoming Control Channel Encryption: Cipher 'AES-256-CTR' initialized with 256 bit key
2025-06-10 17:29:30 Incoming Control Channel Encryption: Using 256 bit message hash 'SHA256' for HMAC authentication
2025-06-10 17:29:30 TCP/UDP: Preserving recently used remote address: [AF_INET]192.168.28.129:1194
2025-06-10 17:29:30 Socket Buffers: R=[212992->212992] S=[212992->212992]
2025-06-10 17:29:30 UDP link local: (not bound)
2025-06-10 17:29:30 UDP link remote: [AF_INET]192.168.28.129:1194
2025-06-10 17:29:30 TLS: Initial packet from [AF_INET]192.168.28.129:1194, sid=739a5206 882daa2e
2025-06-10 17:29:30 VERIFY OK: depth=1, CN=Easy-RSA CA
2025-06-10 17:29:30 VERIFY KU OK
2025-06-10 17:29:30 Validating certificate extended key usage
2025-06-10 17:29:30 ++ Certificate has EKU (str) TLS Web Server Authentication, expects TLS Web Server Authentication
2025-06-10 17:29:30 VERIFY EKU OK
2025-06-10 17:29:30 VERIFY OK: depth=0, CN=server
2025-06-10 17:29:30 WARNING: 'link-mtu' is used inconsistently, local='link-mtu 1500', remote='link-mtu 1601'
2025-06-10 17:29:30 WARNING: 'keysize' is used inconsistently, local='keysize 128', remote='keysize 256'
2025-06-10 17:29:30 Control Channel: TLSv1.3, cipher TLSv1.3 TLS AES 256 GCM SHA384, 2048 bit RSA
2025-06-10 17:29:30 [server] Peer Connection Initiated with [AF_INET]192.168.28.129:1194
2025-06-10 17:29:30 PUSH: Received control message: 'PUSH_REPLY,redirect-gateway def1 bypass-dhcp,dhcp-option DNS 8.8.8.8,dhcp-option DNS 8.8.4.4,block-outside-dns,route-gateway 10.8.0.1,topology subnet,ping 10,ping-restart 120,ifconfig 10.8.0.2 255.255.255.0,peer-id 0,cipher AES-256-GCM'
2025-06-10 17:29:30 Unrecognized option or missing or extra parameter(s) in [PUSH-OPTIONS]:4: block-outside-dns (2.5.1)
2025-06-10 17:29:30 OPTIONS IMPORT: timers and/or timeouts modified
2025-06-10 17:29:30 OPTIONS IMPORT: --ifconfig/up options modified
2025-06-10 17:29:30 OPTIONS IMPORT: route options modified
2025-06-10 17:29:30 OPTIONS IMPORT: route-related options modified
2025-06-10 17:29:30 OPTIONS IMPORT: --ip-win32 and/or --dhcp-option options modified
2025-06-10 17:29:30 OPTIONS IMPORT: peer-id set
2025-06-10 17:29:30 OPTIONS IMPORT: adjusting link-mtu to 1624
2025-06-10 17:29:30 OPTIONS IMPORT: data channel crypto options modified
```

```

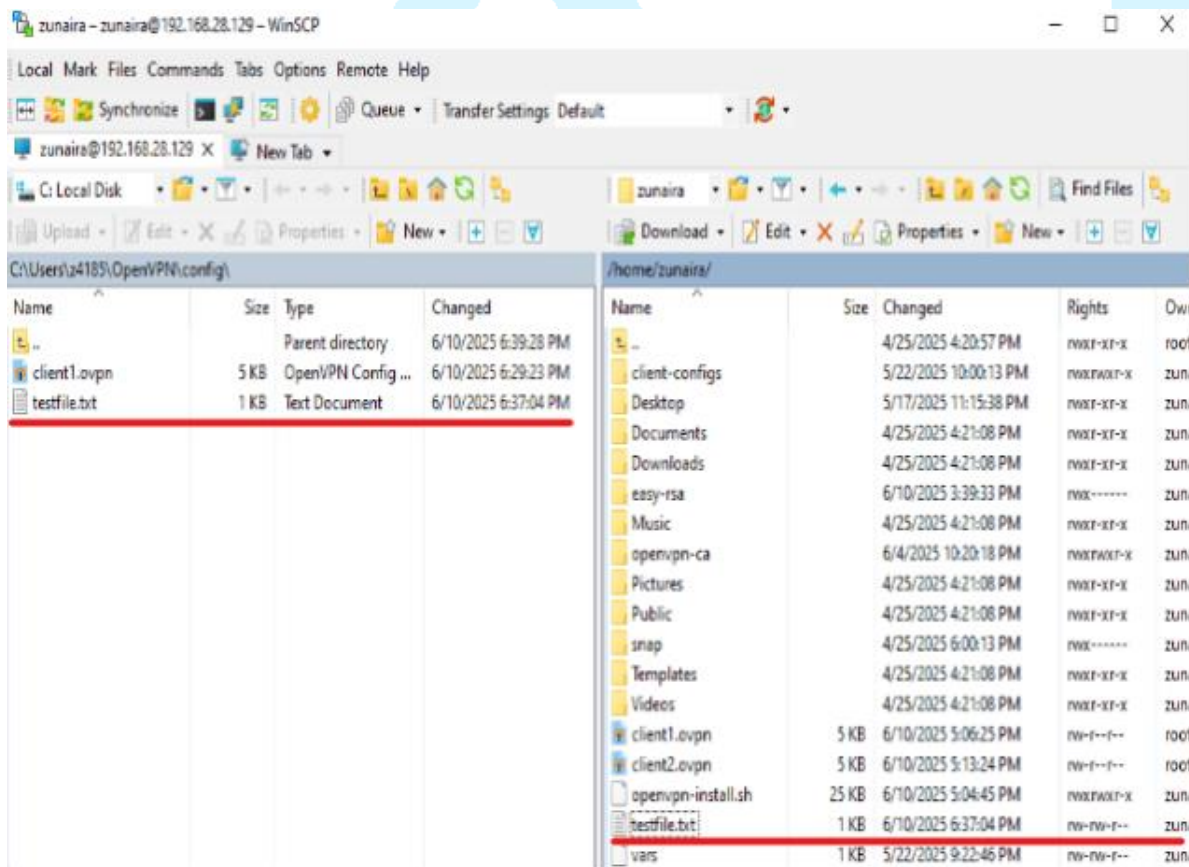
2025-06-10 17:29:30 OPTIONS IMPORT: peer-id set
2025-06-10 17:29:30 OPTIONS IMPORT: adjusting link_mtu to 1624
2025-06-10 17:29:30 OPTIONS IMPORT: data channel crypto options modified
2025-06-10 17:29:30 Data Channel: using negotiated cipher 'AES-256-GCM'
2025-06-10 17:29:30 Outgoing Data Channel: Cipher 'AES-256-GCM' initialized with 256 bit key
2025-06-10 17:29:30 Incoming Data Channel: Cipher 'AES-256-GCM' initialized with 256 bit key
2025-06-10 17:29:30 net_route_v4_best_gw query: dst 0.0.0.0
2025-06-10 17:29:30 net_route_v4_best_gw result: via 192.168.28.2 dev eth0
2025-06-10 17:29:30 ROUTE_GATEWAY 192.168.28.2/255.255.255.0 IFACE=eth0 HWADDR=00:0c:29:77:31:f6
2025-06-10 17:29:30 TUN/TAP device tun0 opened
2025-06-10 17:29:30 net_iface_mtu_set: mtu 1500 for tun0
2025-06-10 17:29:30 net_iface_up: set tun0 up
2025-06-10 17:29:30 net_addr_v4_add: 10.8.0.2/24 dev tun0
2025-06-10 17:29:30 net_route_v4_add: 192.168.28.129/32 via 192.168.28.2 dev eth0 table 0 metric -1
2025-06-10 17:29:30 net_route_v4_add: 0.0.0.0/1 via 10.8.0.1 dev [NULL] table 0 metric -1
2025-06-10 17:29:30 net_route_v4_add: 128.0.0.0/1 via 10.8.0.1 dev [NULL] table 0 metric -1
2025-06-10 17:29:30 WARNING: this configuration may cache passwords in memory -- use the auth-nocache option to prevent this
2025-06-10 17:29:30 Initialization Sequence Completed

```

## PART 2: Transfer Files Between Windows ↔ Ubuntu

This already works using SCP or FileZilla. Confirm it's through the VPN tunnel by checking IP routes (ipconfig / ifconfig).

- Use WinSCP:





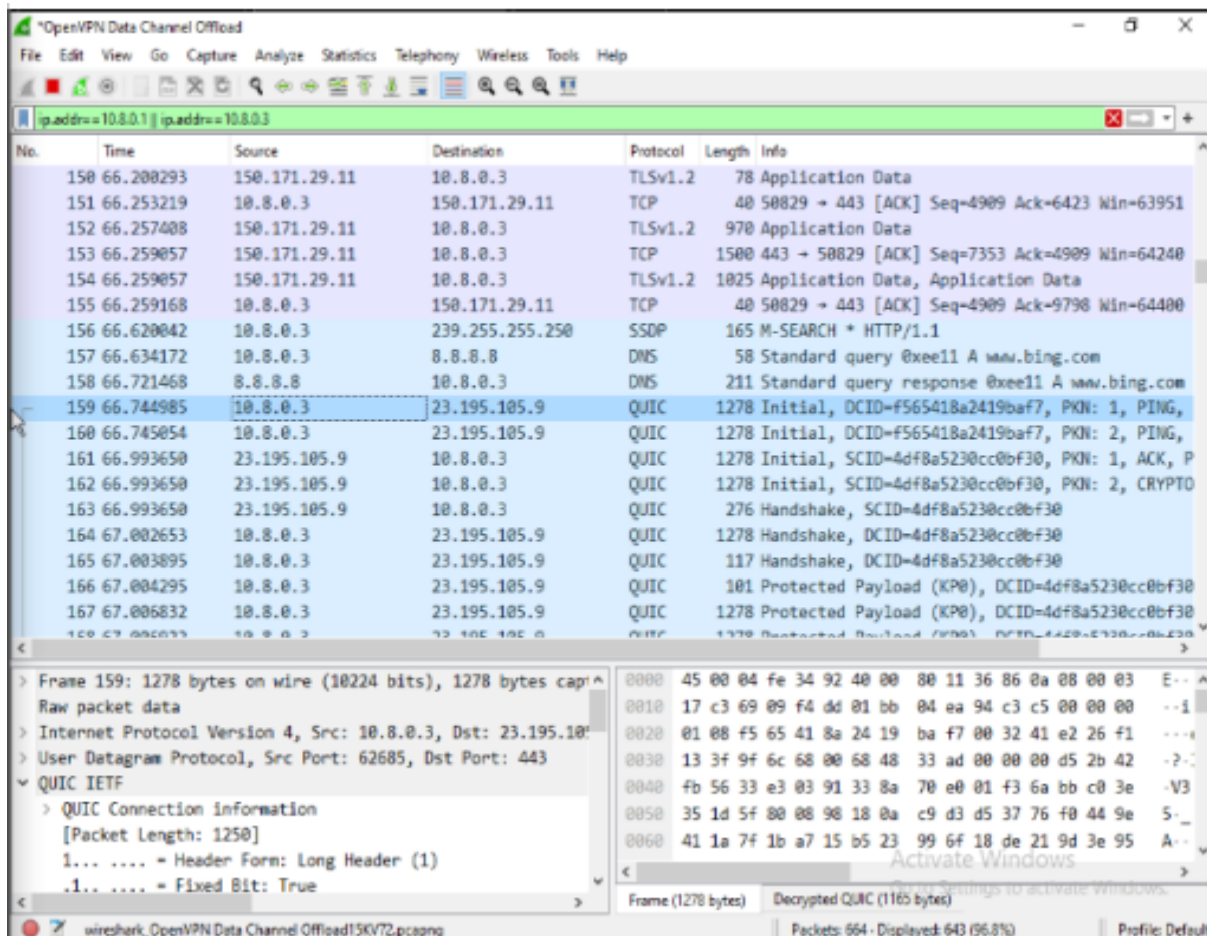
- Verify VPN Route

From Ubuntu:

```
zunaira@zunaira-VMware-Virtual-Platform:~$ sudo tcpdump -i tun0
[sudo] password for zunaira:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
19:21:15.728188 IP 10.8.0.2.59484 > 82.221.107.34.bc.googleusercontent.com.http: Flags [.], ack 843217898, win 64024, length 0
19:21:15.728563 IP 10.8.0.2.59486 > 82.221.107.34.bc.googleusercontent.com.http: Flags [.], ack 1289766690, win 64024, length 0
19:21:15.728718 IP 82.221.107.34.bc.googleusercontent.com.http > 10.8.0.2.59484: Flags [.], ack 1, win 64240, length 0
19:21:15.728957 IP 82.221.107.34.bc.googleusercontent.com.http > 10.8.0.2.59486: Flags [.], ack 1, win 64240, length 0
19:21:17.376066 IP 10.8.0.3.50810 > 172.168.215.153.https: Flags [.], seq 2237298181:2237298182, ack 225331088, win 63436, length 1
19:21:17.685800 IP 10.8.0.3.50806 > a23-54-80-32.deploy.static.akamaitechnologies.com.https: Flags [.], seq 1175108267:1175108268, ack 339227614, win 63626, length 1
19:21:17.685381 IP 10.8.0.3.50807 > a23-54-80-32.deploy.static.akamaitechnologies.com.https: Flags [.], seq 3093294532:3093294533, ack 1244561729, win 64400, length 1
19:21:17.685942 IP a23-54-80-32.deploy.static.akamaitechnologies.com.https > 10.8.0.3.50806: Flags [.], ack 1, win 64240, length 0
```

```
.bing.com.edgekey.net., CNAME e86303.dscx.akamaiedge.net., A 23.195.105.9, A 23.195.105.41 (183)
19:22:28.776305 IP 10.8.0.3.62685 > a23-195-105-9.deploy.static.akamaitechnologies.com.https: UDP, length 1250
19:22:28.776510 IP 10.8.0.3.62685 > a23-195-105-9.deploy.static.akamaitechnologies.com.https: UDP, length 1250
19:22:29.023274 IP a23-195-105-9.deploy.static.akamaitechnologies.com.https > 10.8.0.3.62685: UDP, length 1250
19:22:29.023376 IP a23-195-105-9.deploy.static.akamaitechnologies.com.https > 10.8.0.3.62685: UDP, length 1250
19:22:29.023380 IP a23-195-105-9.deploy.static.akamaitechnologies.com.https > 10.8.0.3.62685: UDP, length 248
19:22:29.036458 IP 10.8.0.3.62685 > a23-195-105-9.deploy.static.akamaitechnologies.com.https: UDP, length 1250
19:22:29.036709 IP 10.8.0.3.62685 > a23-195-105-9.deploy.static.akamaitechnologies.com.https: UDP, length 89
19:22:29.036847 IP 10.8.0.3.62685 > a23-195-105-9.deploy.static.akamaitechnologies.com.https: UDP, length 73
19:22:29.038146 IP 10.8.0.3.62685 > a23-195-105-9.deploy.static.akamaitechnologies.com.https: UDP, length 1250
19:22:29.038425 IP 10.8.0.3.62685 > a23-195-105-9.deploy.static.akamaitechnologies.com.https: UDP, length 1250
19:22:29.038590 IP 10.8.0.3.62685 > a23-195-105-9.deploy.static.akamaitechnologies.com.https: UDP, length 595
^C
241 packets captured
241 packets received by filter
0 packets dropped by kernel
```

From Windows, open Wireshark and filter:



### PART 3: Perform Insider Attack from Parrot

#### Simulate Insider Actions:

- Try scanning Windows from Parrot using nmap

#### Step 1: Scan Windows Using Nmap

```

[~]-(user@parrot)-[~]
$ sudo nmap -sS 10.8.0.3
Starting Nmap 7.91 ( https://nmap.org ) at 2025-06-10 18:56 UTC
Nmap scan report for 10.8.0.3
Host is up (0.0026s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
5357/tcp  open  wsdapi
Nmap done: 1 IP address (1 host up) scanned in 12.26 seconds
  
```

- Attempt unauthorized file access using:

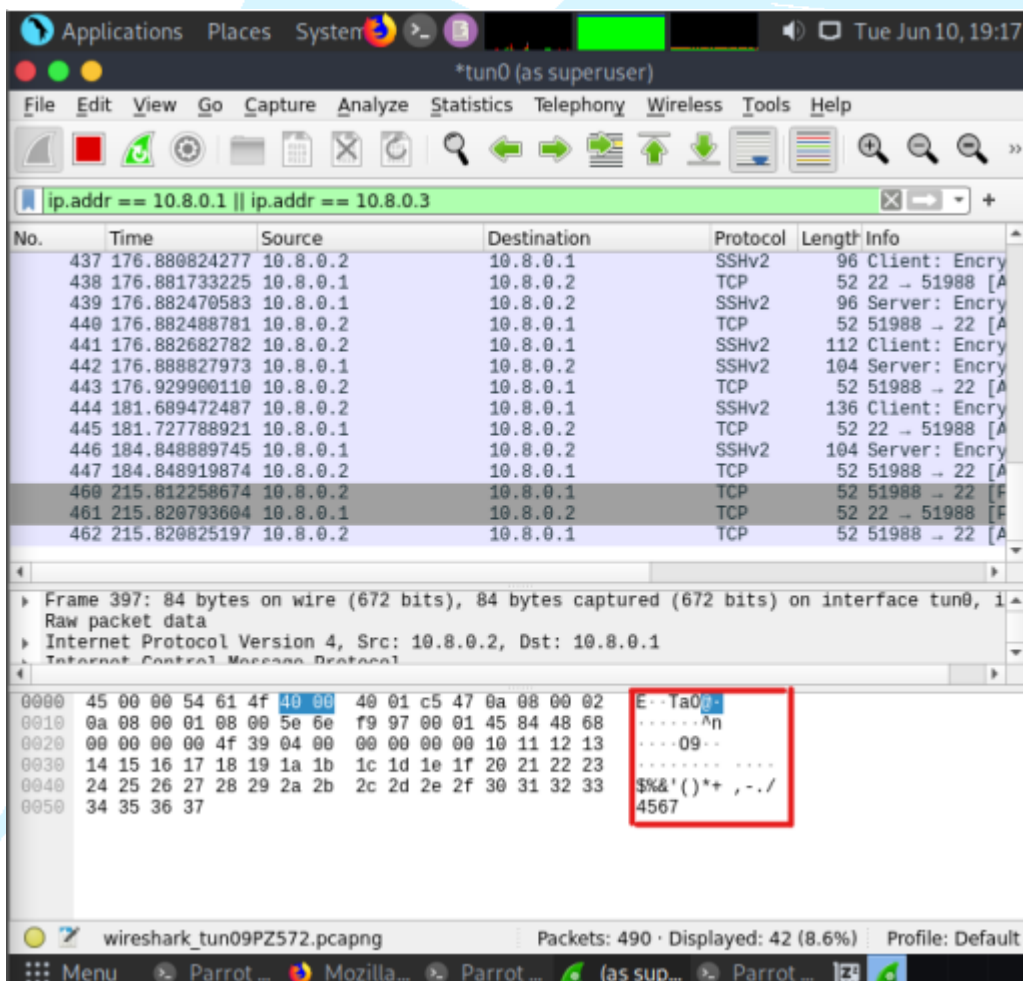
Step 2: Try Accessing Shared Files via SMB

```
(user@parrot)-[~]
$ smbclient //10.8.0.3/c$ -u desktop-cl4mkam\z4185
session setup failed: NT STATUS_ACCESS_DENIED
```

Unauthorized SMB Access Attempt from Parrot (VPN Insider) — Access Denied

- Try sniffing VPN traffic using Wireshark (sudo wireshark)

Step 3: Passive Sniffing with Wireshark



Due to OpenVPN's client-to-server encryption model, peer VPN clients (e.g., Parrot OS) cannot passively sniff traffic between other clients (e.g., Windows ↔ Ubuntu) unless positioned as a gateway or VPN server.

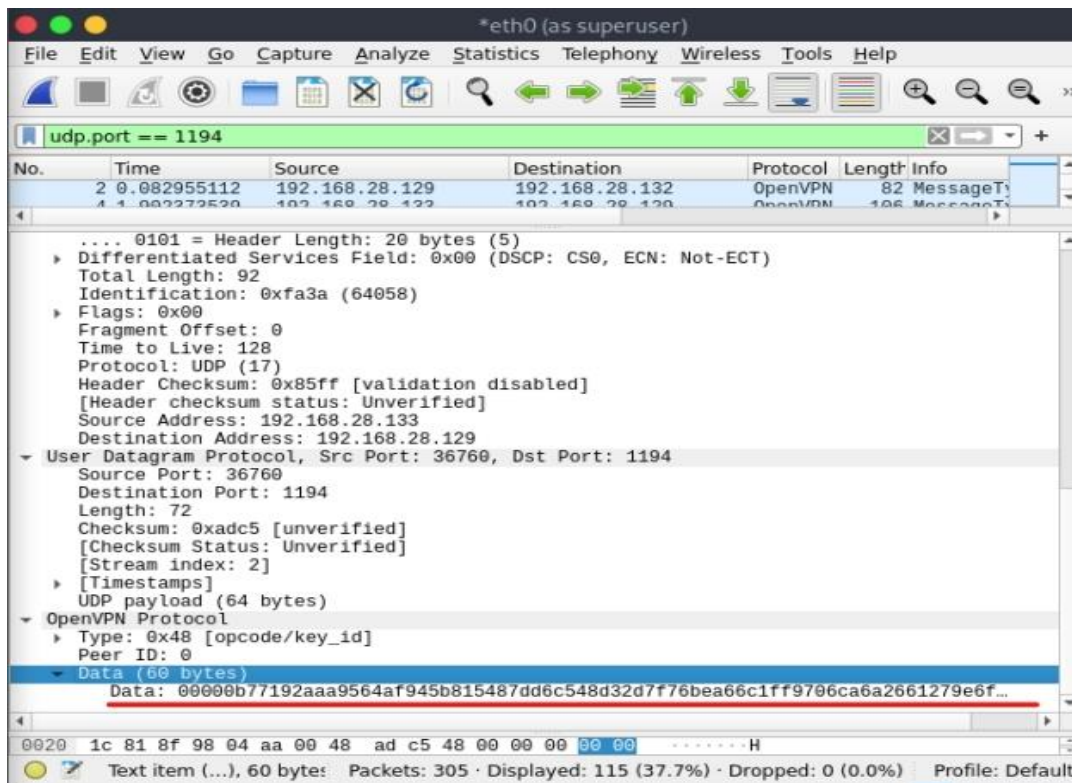






- Look for OpenVPN packets (UDP 1194).

Observe that payload is encrypted – compare with normal LAN traffic.



### Observe Payload

- Under Data, there is a hex dump (right side: random-looking hex bytes)
- The ASCII panel (right-most column) will show unreadable garbage
- No readable text, file names, usernames, or commands

## 1. Capture Decrypted VPN Traffic (Inside the Tunnel)

Wireshark capture on interface `tun0`. The filter is `ip.addr == 10.8.0.1 || ip.addr == 10.8.0.3 || ip.addr == 10.8.0.2`. The packet list shows the following entries:

No.	Time	Source	Destination	Protocol	Length	Info
216	63.549706615	10.8.0.2	34.107.221.82	TCP	40	59670 → 80 [ACK] Seq=300 Ack=217 Win=64024 Len=0
217	64.859928064	34.107.221.82	10.8.0.2	HTTP	256	HTTP/1.1 200 OK (text/plain)
218	64.861366983	10.8.0.2	34.107.221.82	TCP	40	59664 → 80 [RST] Seq=2095 Win=0 Len=0
219	66.355744984	10.8.0.2	34.107.243.93	TLSv1.2	79	Application Data
220	66.356383955	34.107.243.93	10.8.0.2	TCP	40	443 → 53784 [ACK] Seq=313 Ack=352 Win=64240 Len=0
221	66.400362852	34.107.243.93	10.8.0.2	TLSv1.2	79	Application Data
222	66.401663889	10.8.0.2	34.107.243.93	TCP	40	53784 → 443 [ACK] Seq=352 Ack=352 Win=64028 Len=0
223	73.023923233	10.8.0.2	34.107.243.93	TLSv1.2	79	Application Data
224	73.024330500	34.107.243.93	10.8.0.2	TCP	40	443 → 53784 [ACK] Seq=352 Ack=391 Win=64240 Len=0
225	74.524818601	10.8.0.2	34.107.221.82	TCP	40	[TCP Keep-Alive] 59668 → 80 [ACK] Seq=1176 Ack=865
226	74.809872537	10.8.0.2	34.107.221.82	TCP	40	[TCP Keep-Alive] 59670 → 80 [ACK] Seq=299 Ack=217
227	75.662685512	10.8.0.2	34.107.221.82	TCP	40	[TCP Keep-Alive] 59668 → 80 [ACK] Seq=1176 Ack=865
228	75.662936303	34.107.221.82	10.8.0.2	TCP	40	[TCP Keep-Alive ACK] 80 → 59668 [ACK] Seq=865 Ack=
229	75.947792946	10.8.0.2	34.107.221.82	TCP	40	[TCP Keep-Alive] 59670 → 80 [ACK] Seq=299 Ack=217
230	75.948171315	34.107.221.82	10.8.0.2	TCP	40	[TCP Keep-Alive ACK] 80 → 59670 [ACK] Seq=217 Ack=

Packet 221 (79 bytes on wire (632 bits)) is selected. The raw packet data is shown in hexadecimal and ASCII:

```
0000 45 00 00 4f 07 05 00 00 7f 00 14 72 22 6b f3 5d E..0.e...r.k.
0010 8a 00 00 02 01 bb d2 18 13 14 41 00 bc 90 f1 81 .....A....
0020 50 18 fa f0 e5 4e 00 00 17 03 03 00 22 0f 31 6c P...N...."-1l
0030 2d 79 0b 7e aa 12 99 b0 73 e2 35 b9 0e 0f 90 00 y.....s:5.n...
0040 b4 13 55 63 0e 43 57 b3 85 a5 f3 28 55 11 82 ...Uc.Dm...U..
```

## 2. Capture Decrypted VPN Traffic (Inside the Tunnel)

Wireshark capture on interface `ens33`. The filter is `ip.addr == 192.168.28.129 || ip.addr == 192.168.28.132 || ip.addr == 192.168.28.133`. The packet list shows the following entries:

No.	Time	Source	Destination	Protocol	Length	Info
244	105.150111765	192.168.28.133	192.168.28.129	OpenVPN	166	MessageType: P_DATA_V2
245	105.150112025	192.168.28.133	192.168.28.129	OpenVPN	166	MessageType: P_DATA_V2
246	105.150410036	192.168.28.129	34.107.243.93	TCP	54	53808 → 443 [ACK] Seq=1 Ack=25 Win=62760 Len=0
247	105.499854513	192.168.28.133	192.168.28.129	OpenVPN	134	MessageType: P_DATA_V2
248	105.499854756	192.168.28.133	192.168.28.129	OpenVPN	134	MessageType: P_DATA_V2
249	105.500874118	192.168.28.129	34.107.243.93	TLSv1.2	82	Application Data
250	105.500982579	34.107.243.93	192.168.28.129	TCP	60	443 → 53808 [ACK] Seq=25 Ack=29 Win=64240 Len=0
251	105.501580388	192.168.28.129	192.168.28.133	OpenVPN	166	MessageType: P_DATA_V2

Packet 246 (54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface `ens33`, id 0) is selected. The raw packet data is shown in hexadecimal and ASCII:

```
0000 00 50 56 f3 4a c6 00 0c 29 3d c8 be 00 50 56 f3 4a c6
0010 00 28 22 b2 40 00 3f 06 26 2d 79 0b 7e aa 12 99 b0
0020 f3 5d d2 30 01 bb aa a3 82 c0
0030 f5 3c 62 63 00 00
```

## PART 5: Create Security Rules to Prevent Insider Attack

### On Ubuntu (VPN Server):

- UFW firewall rules:

```
zunaira@zunaira-VMware-Virtual-Platform:~$ sudo ufw allow 1194/udp
Skipping adding existing rule
Skipping adding existing rule (v6)
```

```
zunaira@zunaira-VMware-Virtual-Platform:~$ sudo ufw allow ssh
Skipping adding existing rule
Skipping adding existing rule (v6)
```

```
zunaira@zunaira-VMware-Virtual-Platform:~$ sudo ufw enable
Firewall is active and enabled on system startup
```

### Block access between VPN clients (client isolation):

- Edit /etc/openvpn/server.conf:

```
zunaira@zunaira-VMware-Virtual-Platform:~$ nano /etc/openvpn/server.conf
GNU nano 7.2 /etc/openvpn/server.conf *
local 0.0.0.0
proto udp
port 1194
dev tun
server 10.7.8.0 255.255.255.0
topology subnet
ca ca.crt
cert server.crt
key server.key
dh dh.pem
tls-crypt ta.key # CHANGED from tls-auth
data-ciphers AES-256-GCM:AES-128-GCM # CHANGED from cipher
data-ciphers-fallback AES-256-CBC
auth SHA256
user nobody
group nogroup
persist-key
persist-tun
keepalive 10 120
push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DNS 8.8.8.8"
push "dhcp-option DNS 8.8.4.4"
ifconfig-pool-persist /var/log/openvpn/ipp.txt # FIXED path
status /var/log/openvpn/openvpn-status.log
verb 3
explicit-exit-notify 1
#client-to-client
```

## 1. Add specific rules using iptables:

### Block Parrot OS VPN IP from Accessing Windows

- Use iptables:

```
zunaira@zunaira-VMware-Virtual-Platform:~$ sudo iptables -A FORWARD -s 10.8.0.2 -d 10.8.0.3 -j DROP
```

## 2. Restrict file sharing services

```
zunaira@zunaira-VMware-Virtual-Platform:~$ sudo ufw deny from 10.8.0.2 to any port 445
Rule added
zunaira@zunaira-VMware-Virtual-Platform:~$ sudo ufw deny from 10.8.0.2 to any port 139
Rule added
```

```
zunaira@zunaira-VMware-Virtual-Platform:~$ sudo ufw status numbered
Status: active
```

To	Action	From
--	-----	----
[ 1] 1194/udp	ALLOW IN	Anywhere
[ 2] 22/tcp	ALLOW IN	Anywhere
[ 3] OpenSSH	ALLOW IN	Anywhere
[ 4] 445	DENY IN	10.8.0.2
[ 5] 139	DENY IN	10.8.0.2
[ 6] 1194/udp (v6)	ALLOW IN	Anywhere (v6)
[ 7] 22/tcp (v6)	ALLOW IN	Anywhere (v6)
[ 8] OpenSSH (v6)	ALLOW IN	Anywhere (v6)

## Findings & Analysis

Vulnerability	Risk Level	Mitigation	Result
Unencrypted SMB	High	Disabled SMBv1	Access denied
Client-to-client access	Medium	Enabled isolation	Scan blocked
Port scanning	Low	IP filtering	Limited visibility

## Conclusion

### Key Achievements

- Successfully established encrypted VPN tunnel
- Verified payload encryption via Wireshark
- Implemented effective client isolation
- Blocked unauthorized access attempts

### Recommendations

1. Implement two-factor authentication
2. Enable VPN connection logging
3. Regular certificate rotation (90-day policy)

