

# NSR/AS Lab 3 – VPNs

Student Name: Tran Anh Thu Pham  
Student ID: 103818400

TNE30009 – Network Security and  
Resilience  
*Swinburne University of Technology*

103818400@student.swin.edu.au

**Abstract**— This report provides a comprehensive overview of Virtual Private Networks (VPNs), essential tools for secure communication over public networks. Beginning with an introduction to their VPN characteristics, capabilities, and various types including Remote access VPNs, Intranet VPNs, and Extranet VPNs. It highlights key considerations such as security, reliability, and manageability. This report delves into the fundamental concept of tunnelling within VPNs with its components and operations. Through detailed explanations, the role of tunnelling protocols such as IPSec, PPTP, and L2TP in encapsulating and encrypting data packets for secure transmission has been shown. Implementation using OpenVPN is demonstrated through shared key generation, transfer and the establishment of encrypted tunnels between virtual machines. Step-by-step instructions are provided along with insights into monitoring VPN traffic using Wireshark, emphasizing the encryption of data within the VPN tunnel. When comparing the monitored traffic from the tun1 and Ethernet interface, it becomes evident that encryption occurs solely on the tun1 interface after setting up the encrypted tunnel between two virtual machines.

**Keywords**—VPN, OpenVPN, tunnel, shared key, encrypted traffic

## I. INTRODUCTION TO VPNS

### A. VPN characteristics and capabilities

VPN (Virtual Private Network) is defined as a service that makes use of publicly available networking infrastructure to provide the features of a private network. A VPN tunnel is an encrypted connection between a device and a VPN server. VPN Tunneling protocol has responsibilities for encapsulating data packets, IP packets and layer two frames for transmission. Different types of VPN protocols are IP Security (IPSec), Point-to-point tunneling protocol (PPTP), Layer 2 Tunneling Protocol (L2TP), etc. There are three types of VPNs: Remote access VPNs, Intranet VPNs and Extranet VPNs.

General VPN requirements include VPN security, Availability, Reliability, Quality of Service, Compatibility and Manageability. To be more specific, security features such as firewalls, Network Address Translation (NAT), user and packet authentication, as well as encryption are necessary to be implemented to guarantee data transmission. Quality of Service (QoS) includes best effort, and relative, or absolute prioritization to optimise network performance. Availability and reliability ensure the Service Level Agreements (SLAs) from Internet Service Providers (ISPs). Redundant routing, access, and infrastructure from ISPs can enhance reliability. Compatibility requires IP gateways for non-IP traffic such as Frame Relay (FR) and IP tunnelling. Manageability is the ability to manage both remote and local sites.

VPN architectures consist of the following building blocks: VPN hardware (VPN servers, VPN clients, VPN routers, VPN concentrators, IP gateways, Virtual Private Routers, Virtual Private Trunking), VPN software (VPN server software, VPN client software, VPN management software), organisation's security infrastructure (firewalls, authentication systems, IPSec), service provider's security infrastructure (Reliability and availability through redundancy, Security through appropriate hardware and software, supports main tunnelling protocols and access networks), public networks and tunnels.

There are different ways to categorise VPN architectures based on their characteristics. Depending on who implements the VPN (ISP or subscriber), VPN architecture includes Dependent VPNs, Independent VPNs and Hybrid VPNs. Depending on where the VPN endpoints are, there are three types of VPN architecture namely Router to router, Firewall to firewall and Client initiated. Based on the network layer, there are Data Link layer (PPTP, L2TP), Network layer (IPSec) and Transport Layer (SSTP). VPN architectures are categorized based on their size, service and complexity e.g. class 0 for small organizations, class 4 for high speed access options.

VPN implementation brings its users a wide range of benefits in the digital world including low implementation and management costs, enhanced connectivity, robust security, efficient network capacity utilization, scalability, privacy protection and the ability to avoid geoblocking. However, dependence on the reliability of the Internet becomes one of the drawbacks of VPN.

Tunneling is the fundamental aspect of VPN involving encapsulating an entire data packet within the packet of other protocols. Tunneling components include Target network, Initiator node, Home Agent (HA), and Foreign Agent (FA). Tunnel operations include two phases respectively Initiation and Data transfer. In the first phase, the initiator sends the connection request to the Foreign Agent in the local network. Then the FA will authenticate the request. If the authentication is successful, the FA will send the request to the target network Home Agent. If the HA accepts the request, the FA will send the encrypted login ID and password. After the login is verified by the HA, the HA will send a Register Reply message and tunnel ID to the FA. Finally, when the FA receives the Register Reply from HA, the tunnel is established. During the second phase of VPN operations, the initiator begins sending data packets to the FA. Then the FA constructs a tunnel header and a header of a routable protocol, adding them to the data packet. After the tunnel header construction, the FA transmits the resulting routable encrypted packet to the HA using the provided tunnel ID. When the HA receives the encrypted information, the HA removes the tunnel header and routable protocol header. Finally, the original data is then

forwarded to the intended destination node. This process ensures secure and efficient data transmission between two nodes through the VPN tunnel.

A tunnelled packet includes the components illustrated in Fig 1 below.

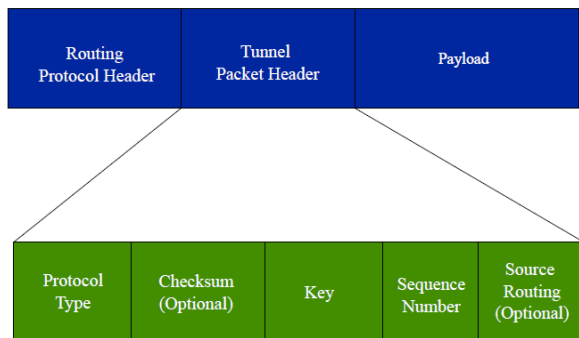


Fig. 1. Tunnelled packet format

A tunnel comprises three protocols: The carrier protocol, the encapsulating protocol, and the passenger protocol. To be more specific, the carrier protocol is responsible for routing tunnelled packets across the network, typically using IP. The encapsulating protocol adds necessary headers for secure transmission and the passenger protocol encapsulates the original data, often using PPP or IP.

There are two types of tunnels: Voluntary tunnels and compulsory tunnels. The key difference between these two types is their initiation and purpose. Voluntary tunnels are established end to end at the request of one of the endpoints. In contrast, compulsory tunnels are created and configured by an intermediate device. Voluntary tunnels serve the exclusive use of a single communication while compulsory tunnels are shared by multiple communications.

### B. VPNs implementation in Organisational security policy

Implementing VPNs in organizational security policy safeguards sensitive data and ensures secure Internet communication. The organisations should point out their overall security objectives to specify who can use VPNs, for what purposes, and under what circumstances. The clear Organization's security policy is then defined to align with these requirements.

Access to VPNs is strictly controlled by implementing strong authentication mechanisms to restrict access to authorized people only. Some types of VPN authentication methods are Password-based authentication, multi-factor authentication (MFA), and Role-based access control (RBAC).

VPN connections are required to use strong encryption protocols such as AES (Advanced Encryption Standard) to protect data transmitted over that network. AES provides a high level of cryptographic strength of resistance against brute-force attacks. It operates by transforming plaintext data into ciphertext using a symmetric key algorithm, where the same secret key is used for both encryption and decryption processes. The symmetric key is securely exchanged between the VPN client and server during the establishment of the VPN tunnel.

Logging and monitoring mechanisms should be implemented to track VPN usage and detect any suspicious

activities. Monitor VPN connections for signs of unauthorized access or potential security breaches. The monitoring strategy encompasses real-time surveillance and analysis of VPN traffic patterns, anomalous behaviours, and security events using intrusion detection systems (IDS).

All devices connecting to the VPN are required to adhere to minimum security standards. The deployment of endpoint protection solutions includes antivirus software, anti-malware scanners, and firewalls. Additionally, endpoint detection and response (EDR) should be employed to provide real-time visibility into endpoint activities, identify suspicious behaviours, and respond to advanced threats proactively.

To mitigate the risk of data leakage and insider threats, endpoint data protection measures such as data loss prevention (DLP) controls to safeguard sensitive information stored on endpoint devices. This includes encrypting data using full disk encryption (FDE) or file-level encryption to prevent unauthorized access to confidential data in the event of device theft or loss.

## II. OPENVPN BEHAVIOUR

### A. Preliminary

In the lab, a Linux software called OpenVPN is used to set up an encrypted VPN tunnel between two Virtual Machines (VMs). OpenVPN uses Transport Layer Security (TLS) for authentication and negotiation of an encryption algorithm. This process starts with the Ubuntu machine setups and host configuration. After downloading the two machines named VM1 and VM2, the IPv4 addresses of these machines must be identified by typing the command: *ifconfig* in the terminals.

The ethernet interface (enp0s3) should list the IPv4 address of the host after running the above command (Fig. 2). In this case, the IP address of VM1 is: 192.168.65.129 and VM2 is: 192.168.65.128

```
nsr@nsr-VirtualBox: ~/Desktop
nsr@nsr-VirtualBox:~/Desktop$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.65.128 netmask 255.255.255.0 broadcast 192.168.65.255
    inet6 fe80::s7a9:c57a:71a1:929b prefixlen 64 scopeid 0x20<link>
    ether 08:0c:29:be:d8:dd txqueuelen 1000 (Ethernet)
    RX packets 310 bytes 321486 (321.4 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 176 bytes 19211 (19.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 65 bytes 7219 (7.2 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 65 bytes 7219 (7.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Fig. 2. IP address of VM2 in Ethernet interface section

To check the connectivity between two hosts, type the following command on the VM2 terminal to ping VM1 from VM2:

*ping 192.168.65.129*

It is similar to ping VM2 from VM1. If two hosts can ping each other, the results should appear as follows (Fig. 3).

```
nsr@nsr-VirtualBox:~/Desktop$ ping 192.168.65.129
PING 192.168.65.129 (192.168.65.129) 56(84) bytes of data.
64 bytes from 192.168.65.129: icmp_seq=1 ttl=64 time=0.260 ms
64 bytes from 192.168.65.129: icmp_seq=2 ttl=64 time=0.837 ms
64 bytes from 192.168.65.129: icmp_seq=3 ttl=64 time=0.276 ms
64 bytes from 192.168.65.129: icmp_seq=4 ttl=64 time=0.287 ms
64 bytes from 192.168.65.129: icmp_seq=5 ttl=64 time=0.875 ms
64 bytes from 192.168.65.129: icmp_seq=6 ttl=64 time=0.808 ms
64 bytes from 192.168.65.129: icmp_seq=7 ttl=64 time=0.470 ms
64 bytes from 192.168.65.129: icmp_seq=8 ttl=64 time=0.868 ms
64 bytes from 192.168.65.129: icmp_seq=9 ttl=64 time=0.824 ms
64 bytes from 192.168.65.129: icmp_seq=10 ttl=64 time=0.824 ms
64 bytes from 192.168.65.129: icmp_seq=11 ttl=64 time=0.895 ms
64 bytes from 192.168.65.129: icmp_seq=12 ttl=64 time=0.829 ms
64 bytes from 192.168.65.129: icmp_seq=13 ttl=64 time=0.871 ms
^C
--- 192.168.65.129 ping statistics ---
13 packets transmitted, 13 received, 0% packet loss, time 12138ms
rtt min/avg/max/mdev = 0.260/0.686/0.895/0.247 ms
nsr@nsr-VirtualBox:~/Desktop$
```

Fig. 3. Successful ping from VM2 to VM1

When needing to stop the ping, press ctrl-c.

### B. Shared password generation

After verifying that the connection between two hosts is established, the shared password should be generated on VM2. In the command prompt Window, type:

```
Openvpn --genkey --secret mykey
```

The file called “mykey” will be created. To verify that the file is created or not, type the following command:

```
ls -l
```

This file should be listed as Fig. .

```
rtt min/avg/max/mdev = 0.260/0.686/0.895/0.247 ms
nsr@nsr-VirtualBox:~/Desktop$ openvpn --genkey --secret mykey
nsr@nsr-VirtualBox:~/Desktop$ ls -l
total 4
-rw----- 1 nsr nsr 636 Apr  9 20:19 mykey
nsr@nsr-VirtualBox:~/Desktop$
```

Fig. 4. Verification of the creation of mykey file on VM2

The file can also be seen on VM’s Desktop (Fig. 5).

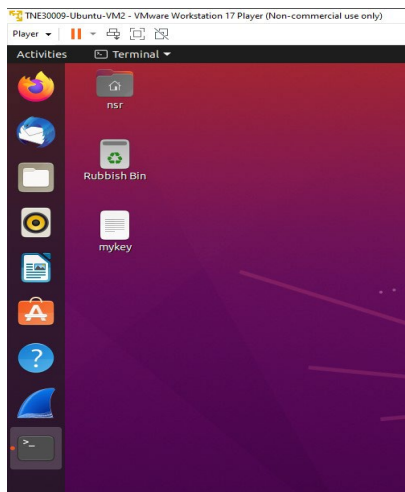


Fig. 5. Shared key file named mykey on VM’s Desktop

To see the contents in the file, type the following command:

```
cat mykey
```

The contents of the file should be listed in Fig. 5.

```
nsr@nsr-VirtualBox:~/Desktop$ cat mykey
#
# 2048 bit OpenVPN static key
#
-----BEGIN OpenVPN Static key V1-----
33205bed975144ba37037cfb19229082
09654fb9d0bbd703e0647f825a9e6ef4
63879c68cea6596e259634359a692847
45ab9963180cf2f5f29bf020ce71e0c2
0dd18e6bba75036e2e5b527fab9d6edf
32c1f9602d2b35790b4122ffc2b796f3
b1c46141b7f2d097cc5dc7d32c8b5246
f4bc1fc94c8e057ec2936c1f54292927
d9f399b7310859cec983e561bbd1a77f
2b6b4783d20ec5313ec895ae5020e3ca
1822e800dbcc4e1b2840cacbbf739123
475221696ab63f903551ac5c97c77336d
60a7f59d37c83e937332a13938c0f072
12ffbdac667e0d43de1372353866114c
a1fe0c8f63fd2daa8ace960de8595483
1ebdcccddc1479c9ad8a0e0d90879ef60
-----END OpenVPN Static key V1-----
nsr@nsr-VirtualBox:~/Desktop$
```

Fig. 6. mykey file contents listed

Now the shared key is created successfully on VM2.

### C. Shared key transferred to the other machine

To transfer the shared key created in VM2 to VM1, Secure Copy (scp) is used on VM1 to copy the key from VM2. At the end of the process, both machines have the same file with the same contents.

The first step of this process is installing ssh which includes scp on each host. Type the following commands on both machines:

```
Sudo apt update
```

```
Sudo apt-get install openssh-server
```

The process of installing SSH is shown in Fig. 6.

```
nsr@nsr-VirtualBox:~/Desktop$
nsr@nsr-VirtualBox:~/Desktop$ sudo apt-get install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  ncurses-term openssh-client openssh-sftp-server ssh-import-id
Suggested packages:
  keychain libpam-ssh monkeysphere ssh-askpass molly-guard
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
The following packages will be upgraded:
  openssh-client
1 to upgrade, 4 to newly install, 0 to remove and 652 not to upgrade.
Need to get 1,359 kB of archives.
After this operation, 6,027 kB of additional disk space will be used.
Do you want to continue? [Y/n] yes
Get:1 http://au.archive.ubuntu.com/ubuntu focal-updates/main amd64 openssh-clie
nt amd64 1:8.2p1-4ubuntu0.11 [670 kB]
Get:2 http://au.archive.ubuntu.com/ubuntu focal-updates/main amd64 ncurses-ter
m all 6.2-0ubuntu2.1 [249 kB]
Get:3 http://au.archive.ubuntu.com/ubuntu focal-updates/main amd64 openssh-sft
p-server amd64 1:8.2p1-4ubuntu0.11 [51.7 kB]
debconf: delaying package configuration, since apt-utils is not installed
Fetched 1,359 kB in 1s (1,359 kB/s)
Selecting previously unselected package ncurses-term.
(Reading database ... 123456789 files and directories currently installed.)
Preparing to unpack .../ncurses-term_6.2-0ubuntu2.1_all.deb ...
Unpacking ncurses-term (6.2-0ubuntu2.1) ...
Selecting previously unselected package openssh-client.
Preparing to unpack .../openssh-client_1:8.2p1-4ubuntu0.11_amd64.deb ...
Unpacking openssh-client (1:8.2p1-4ubuntu0.11) ...
Selecting previously unselected package openssh-sftp-server.
Preparing to unpack .../openssh-sftp-server_1:8.2p1-4ubuntu0.11_amd64.deb ...
Unpacking openssh-sftp-server (1:8.2p1-4ubuntu0.11) ...
Selecting previously unselected package openssh-server.
Preparing to unpack .../openssh-server_1:8.2p1-4ubuntu0.11_amd64.deb ...
Unpacking openssh-server (1:8.2p1-4ubuntu0.11) ...
Setting up ncurses-term (6.2-0ubuntu2.1) ...
Setting up openssh-client (1:8.2p1-4ubuntu0.11) ...
Setting up openssh-sftp-server (1:8.2p1-4ubuntu0.11) ...
Setting up openssh-server (1:8.2p1-4ubuntu0.11) ...
nsr@nsr-VirtualBox:~/Desktop$
```

Fig. 7. ssh installation

The next step of this process is going back to the machine that generated the key previously, in this case, VM2. On VM2, type the following command (Fig. 8).

```
scp mykey nsr@192.168.65.129:Desktop/mykey
```

```
nsr@nsr-VirtualBox:~/Desktop$ scp mykey nsr@192.168.65.129:Desktop/mykey
The authenticity of host '192.168.65.129 (192.168.65.129)' can't be established
ECDSA key fingerprint is SHA256:T2FGE/jY7t+1tDLCLtV6MhyPSEl3JtDhe73ICRgnhMI.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.65.129' (ECDSA) to the list of known hosts.
nsr@192.168.65.129's password:
Permission denied, please try again.
nsr@192.168.65.129's password:
Permission denied, please try again.
nsr@192.168.65.129's password:
mykey
100% 636 823.5KB/s 00:00
nsr@nsr-VirtualBox:~/Desktop$
nsr@nsr-VirtualBox:~/Desktop$
```

Fig. 8. Copy mykey file from VM2 to VM1



Note that the IP address in the above command is the IP address of the VM which copies the key from the VM that created the key before. In this case, the IP address of VM1.

After completing all the steps above, check if the shared key generated from VM2 is also copied to VM1 or not. To do this verification, type the following commands on VM1:

```
ls -l
```

```
cat mykey
```

The mykey file and its contents should be listed on the VM1 terminal similarly to Fig. 4 and Fig. 6.

#### D. Set up an encrypted tunnel

After the shared key are on both machines, the encrypted tunnel should be set up.

VM1 IP address: 192.168.65.129

VM2 IP address: 192.168.65.128

To set up the tunnel from VM1 to VM2 with the IP address of the tunnel endpoints 10.4.0.1 and 10.4.0.2 for VM1 and VM2 respectively, type the following command on VM1 (Fig. 9).

```
sudo openvpn --remote 192.168.65.128 --dev tun1 --ifconfig 10.4.0.1 10.4.0.2 --verb 5 --secret mykey
```

```
nsr@nsr-VirtualBox:~/Desktop$ sudo openvpn --remote 192.168.65.128 --dev tun1 --ifconfig 10.4.0.1 10.4.0.2 --verb 5 --secret mykey
Tue Apr 9 21:07:19 2024 us=780993 disabling NCP mode (--ncp-disable) because not in P2MP client or server mode
Tue Apr 9 21:07:19 2024 us=780981 Current Parameter Settings:
Tue Apr 9 21:07:19 2024 us=780727 config = '[UNDEF]'
Tue Apr 9 21:07:19 2024 us=780736 mode = 0
Tue Apr 9 21:07:19 2024 us=780736 persist_config = DISABLED
Tue Apr 9 21:07:19 2024 us=780737 persist_mode = 1
Tue Apr 9 21:07:19 2024 us=780736 show_ciphers = DISABLED
Tue Apr 9 21:07:19 2024 us=780736 show_digests = DISABLED
Tue Apr 9 21:07:19 2024 us=780737 show_engines = DISABLED
Tue Apr 9 21:07:19 2024 us=780736 genkey = DISABLED
Tue Apr 9 21:07:19 2024 us=780721 key_pass_file = '[UNDEF]'
Tue Apr 9 21:07:19 2024 us=780723 show_tls_ciphers = DISABLED
Tue Apr 9 21:07:19 2024 us=780724 connect_retry_max = 0
Tue Apr 9 21:07:19 2024 us=780723 Connection profiles [0]:
Tue Apr 9 21:07:19 2024 us=780738 proto = udp
Tue Apr 9 21:07:19 2024 us=780725 local_port = '1194'
Tue Apr 9 21:07:19 2024 us=780725 remote = '192.168.65.128'
Tue Apr 9 21:07:19 2024 us=780725 remote_port = '1194'
Tue Apr 9 21:07:19 2024 us=780727 remote_float = DISABLED
Tue Apr 9 21:07:19 2024 us=780726 bind_defined = DISABLED
Tue Apr 9 21:07:19 2024 us=780726 bind_local = ENABLED
Tue Apr 9 21:07:19 2024 us=780729 bind_ipv6_only = DISABLED
Tue Apr 9 21:07:19 2024 us=780730 connect_retry_seconds = 5
Tue Apr 9 21:07:19 2024 us=780730 connect_timeout = 120
Tue Apr 9 21:07:19 2024 us=780732 socks_proxy_server = '[UNDEF]'
Tue Apr 9 21:07:19 2024 us=780732 socks_proxy_port = '[UNDEF]'
Tue Apr 9 21:07:19 2024 us=780732 tun_mtu = 1500
Tue Apr 9 21:07:19 2024 us=780733 tun_mtu_defined = ENABLED
Tue Apr 9 21:07:19 2024 us=780733 link_mtu = 1500
Tue Apr 9 21:07:19 2024 us=780734 link_mtu_defined = DISABLED
Tue Apr 9 21:07:19 2024 us=780733 tun_mtu_extra = 0
Tue Apr 9 21:07:19 2024 us=780734 tun_mtu_extra_defined = DISABLED
Tue Apr 9 21:07:19 2024 us=780736 mtu_discover_type = 1
Tue Apr 9 21:07:19 2024 us=780737 fragment = 0
Tue Apr 9 21:07:19 2024 us=780736 mssfix = 1500
Tue Apr 9 21:07:19 2024 us=780786 explicit_exit_notification = 0
Tue Apr 9 21:07:19 2024 us=780733 Connection profiles END
Tue Apr 9 21:07:19 2024 us=780740 remote_random = DISABLED
Tue Apr 9 21:07:19 2024 us=780740 tpmchange = '[UNDEF]'
Tue Apr 9 21:07:19 2024 us=780744 dev = 'tun1'
Tue Apr 9 21:07:19 2024 us=780741 dev_node = '[UNDEF]'
Tue Apr 9 21:07:19 2024 us=780742 dev_type = '[UNDEF]'
Tue Apr 9 21:07:19 2024 us=780743 l3addr = '[UNDEF]'
Tue Apr 9 21:07:19 2024 us=780744 topology = 1
Tue Apr 9 21:07:19 2024 us=780743 ifconfig_local = '10.4.0.1'
Tue Apr 9 21:07:19 2024 us=780742 ifconfig_remote_netmask = '10.4.0.2'
Tue Apr 9 21:07:19 2024 us=780746 ifconfig_netmask = DISABLED
Tue Apr 9 21:07:19 2024 us=780745 ifconfig_net6 = DISABLED
Tue Apr 9 21:07:19 2024 us=780745 ifconfig_tun6_local = '[UNDEF]'
```

Fig. 9. Set up the encrypted tunnel from VM1 to VM2

This command sets up a tunnel called tun1 from VM1 to VM2. The IP addresses of the tunnel endpoints are 10.4.0.1 on VM1 and 10.4.0.2 on VM2. The remote address is the IP address of VM2 and the traffic going via the tunnel is encrypted using the key contained in the file “mykey”.

After that, the tunnel will be set up from VM2 back to VM1 with the remote address as the IP address of VM1, type the following command on VM2 (Fig. 10).

```
Sudo openvpn --remote 192.168.65.129 --dev tun1 --ifconfig 10.4.0.2 10.4.0.1 --verb 5 --secret mykey
```

```
nsr@nsr-VirtualBox:~/Desktop$ sudo openvpn --remote 192.168.65.129 --dev tun1 --ifconfig 10.4.0.2 10.4.0.1 --verb 5 --secret mykey
(Sudo) password for nsr:
Tue Apr 9 21:00:25 2024 us=794557 disabling NCP mode (--ncp-disable) because not in P2MP client or server mode
Tue Apr 9 21:00:25 2024 us=794465 Current Parameter Settings:
Tue Apr 9 21:00:25 2024 us=794783 config = '[UNDEF]'
Tue Apr 9 21:00:25 2024 us=794736 mode = 0
Tue Apr 9 21:00:25 2024 us=794769 persist_config = DISABLED
Tue Apr 9 21:00:25 2024 us=794801 persist_mode = 1
Tue Apr 9 21:00:25 2024 us=794833 show_ciphers = DISABLED
Tue Apr 9 21:00:25 2024 us=794834 show_digests = DISABLED
Tue Apr 9 21:00:25 2024 us=794856 show_engines = DISABLED
Tue Apr 9 21:00:25 2024 us=794932 genkey = DISABLED
Tue Apr 9 21:00:25 2024 us=794987 key_pass_file = '[UNDEF]'
Tue Apr 9 21:00:25 2024 us=795019 show_tls_ciphers = DISABLED
Tue Apr 9 21:00:25 2024 us=795051 connect_retry_max = 0
Tue Apr 9 21:00:25 2024 us=795084 Connection profiles [0]:
Tue Apr 9 21:00:25 2024 us=795116 proto = udp
Tue Apr 9 21:00:25 2024 us=795148 local_port = '[UNDEF]'
Tue Apr 9 21:00:25 2024 us=795180 local_port = '1194'
Tue Apr 9 21:00:25 2024 us=795212 remote = '192.168.65.129'
Tue Apr 9 21:00:25 2024 us=795244 remote_port = '1194'
Tue Apr 9 21:00:25 2024 us=795275 remote_float = DISABLED
Tue Apr 9 21:00:25 2024 us=795307 bind_defined = DISABLED
Tue Apr 9 21:00:25 2024 us=795339 bind_local = ENABLED
Tue Apr 9 21:00:25 2024 us=795370 bind_ipv6_only = DISABLED
Tue Apr 9 21:00:25 2024 us=795406 connect_retry_seconds = 5
Tue Apr 9 21:00:25 2024 us=795456 connect_timeout = 120
Tue Apr 9 21:00:25 2024 us=795490 socks_proxy_server = '[UNDEF]'
Tue Apr 9 21:00:25 2024 us=795522 socks_proxy_port = '[UNDEF]'
Tue Apr 9 21:00:25 2024 us=795554 tun_mtu = 1500
Tue Apr 9 21:00:25 2024 us=795586 tun_mtu_defined = ENABLED
Tue Apr 9 21:00:25 2024 us=795618 link_mtu = 1500
Tue Apr 9 21:00:25 2024 us=795650 link_mtu_defined = DISABLED
Tue Apr 9 21:00:25 2024 us=795682 tun_mtu_extra = 0
Tue Apr 9 21:00:25 2024 us=795713 tun_mtu_extra_defined = DISABLED
Tue Apr 9 21:00:25 2024 us=795745 mtu_discover_type = 1
Tue Apr 9 21:00:25 2024 us=795777 fragment = 0
Tue Apr 9 21:00:25 2024 us=795810 mssfix = 1500
Tue Apr 9 21:00:25 2024 us=795833 explicit_exit_notification = 0
Tue Apr 9 21:00:25 2024 us=795842 Connection profiles END
Tue Apr 9 21:00:25 2024 us=795859 remote_random = DISABLED
Tue Apr 9 21:00:25 2024 us=795857 tpmchange = '[UNDEF]'
Tue Apr 9 21:00:25 2024 us=795865 dev = 'tun1'
Tue Apr 9 21:00:25 2024 us=795872 dev_type = '[UNDEF]'
Tue Apr 9 21:00:25 2024 us=795886 dev_node = '[UNDEF]'
Tue Apr 9 21:00:25 2024 us=795887 l3addr = '[UNDEF]'
Tue Apr 9 21:00:25 2024 us=795894 topology = 1
```

Fig. 10. Set up the encrypted tunnel from VM2 to VM1

To verify the encrypted bi-directional tunnel between both VMs, on both VMs, type the following command on the other terminals:

```
ifconfig
```

After pressing enter, a new interface called “tun1” will appear which is the encrypted tunnel created previously in terminals of VM1 (Fig. 11) and VM2 (Fig. 12).

```
nsr@nsr-VirtualBox:~/Desktop$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.65.129 netmask 255.255.255.0 broadcast 192.168.65.255
    inet6 fe80::50b7:3bb2:3b2e: prefixlen 64 scopeid 0x20<link>
    ether 08:00:c2:9f:62:40 txqueuelen 1000 (Ethernet)
    RX packets 288080 bytes 429298274 (429.2 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 25096 bytes 1608671 (1.6 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 360 bytes 43601 (43.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 360 bytes 43601 (43.6 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

tun1: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 10.4.0.1 netmask 255.255.255.255 destination 10.4.0.2
    inet6 fe80::a726:5d86:3d21: prefixlen 64 scopeid 0x20<link>
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 100 (UNSPEC)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3 bytes 144 (144.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

nsr@nsr-VirtualBox:~/Desktop$
```

Fig. 11. tun1 interface appeared after setting up the encrypted tunnel on VM1

```
nsr@nsr-VirtualBox:~/Desktop$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.65.128 netmask 255.255.255.0 broadcast 192.168.65.255
    inet6 fe80::a7a9:c57a:71a1:9290 prefixlen 64 scopeid 0x20<link>
    ether 08:00:c2:9f:62:40 txqueuelen 1000 (Ethernet)
    RX packets 278685 bytes 416163979 (416.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 26657 bytes 1696198 (1.6 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 371 bytes 46011 (46.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 371 bytes 46011 (46.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

tun1: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 10.4.0.2 netmask 255.255.255.255 destination 10.4.0.1
    inet6 fe80::81b4:cbac:99f2:71c prefixlen 64 scopeid 0x20<link>
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 100 (UNSPEC)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3 bytes 144 (144.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

nsr@nsr-VirtualBox:~/Desktop$
```

Fig. 12. tun1 interface appeared after setting up the encrypted tunnel on VM2

#### E. VPN tunnel monitoring on Wireshark

On VM2, ping the destination address of the tunnel, in this case, 10.4.0.1 (VM1 endpoint address).

ping 10.4.0.1

On Wireshark of VM2, the traffic from tun1 is captured when starting ping the destination address of the tunnel (Fig. 13).

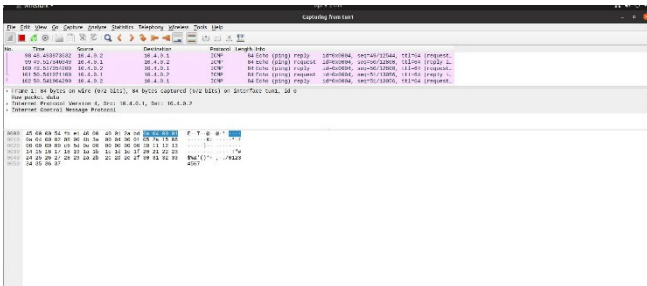


Fig. 13. Encrypted traffic captured from tun1 on VM2 Wireshark

On Wireshark of VM2, the traffic from the ethernet interface is captured when starting ping the destination address of the tunnel (Fig. 14).

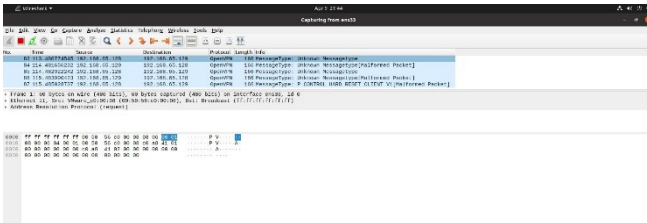


Fig. 14. Encrypted traffic captured from ens33 on VM2 Wireshark

After the telnet, monitoring the traffic captured from tun1 and ethernet interface respectively (Fig. 16 and Fig. 17).

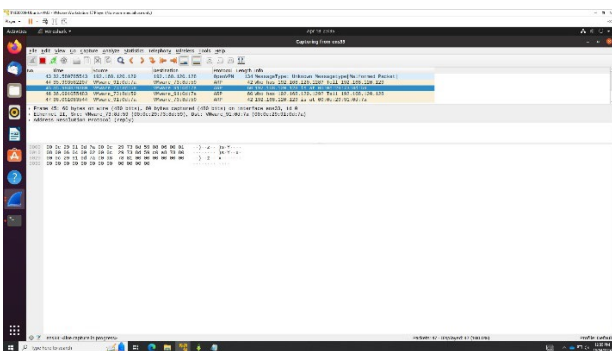


Fig. 16. Traffic captured from ethernet interface when using telnet

When monitoring the traffic from these two interfaces, tun1 and ens33, the traffic is only encrypted when captured from tun1. The traffic is encrypted when captured from tun1 because the VPN tunnel was established using OpenVPN with the same shared key on both machines. This VPN tunnel allows traffic to be encrypted when travelling through the VPN connection. The reason why the traffic is not encrypted when captured from the ethernet interface is because traffic on the ethernet interface is the original, unencrypted data before it enters the VPN tunnel. In another way, the VPN tunnel encrypts data before sending it over the physical network.

To telnet from VM2 to VM1, on VM2 type the command (Fig. 15):

telnet 10.4.0.1

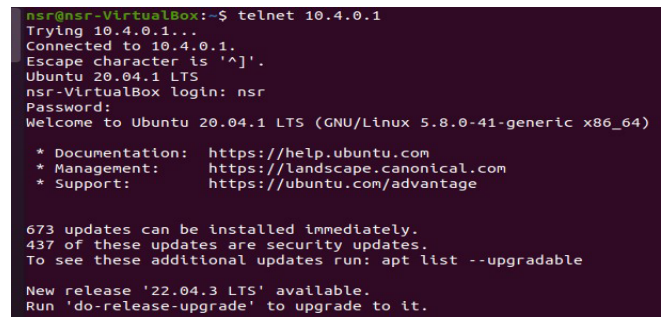


Fig. 15. Telnet from VM2 to VM1

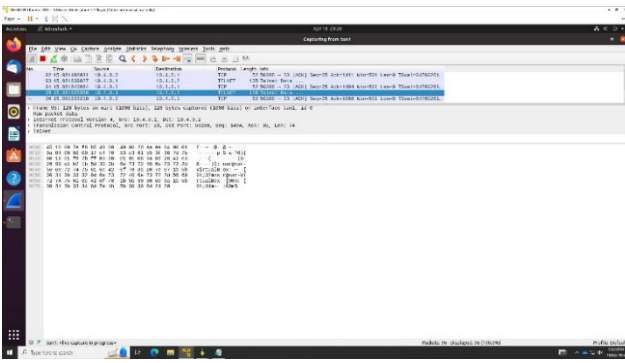


Fig. 17. Traffic captured from tun1 interface when using telnet

After using telnet, the tun1 interface is not encrypted, and traffic on the Ethernet interface is encrypted using the OpenVPN SSL protocol. The encryption status of the traffic on the tun1 interface depends on whether the application used for communication within the VPN tunnel has its encryption mechanism. In this case, telnet is used, which does not have encryption, the traffic would be fully decrypted on the tun1 interface.

### III. CONCLUSION

In conclusion, this report has provided a comprehensive exploration of Virtual Private Networks (VPNs), and a description of their role in network security. By delving into VPNs' characteristics, types, architectures of VPNs, their essential components and considerations involved in their implementation. From security and reliability to manageability and scalability, VPNs offer a robust solution for ensuring communication over untrusted networks.

Furthermore, VPN setup using OpenVPN has been demonstrated through shared key generation, transfer, and the establishment of encrypted tunnels between two virtual machines. Additionally, the monitoring of VPN traffic

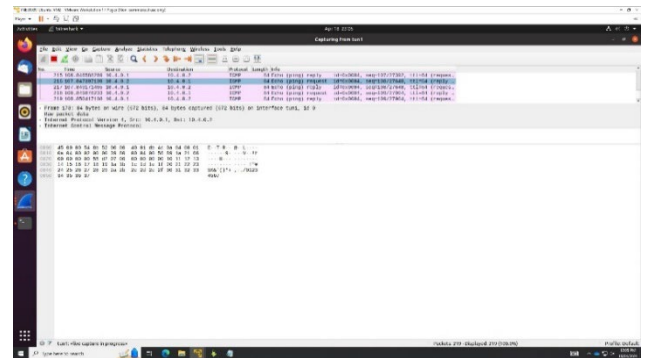


Fig. 18. Traffic captured from tun1 interface before using telnet

At the end of the lab, the question “What is a VPN tunnel?” can be answered: It is a secure, encrypted communication pathway established over a public or untrusted network, enabling secure transmission of data between two endpoints. In this lab, the VPN tunnel was created using OpenVPN between VM1 (an IP address is 192.168.65.129 with remote address 10.4.0.1) and VM2 (an IP address is 192.168.65.128 with remote address 10.4.0.2).

using Wireshark has highlighted the encryption mechanism employed within VPN tunnels.

#### REFERENCES

- [1]“Redirecting,” login.microsoftonline.com.  
[https://swinburne.instructure.com/courses/57020/pages/lect-ures-week-5?module\\_item\\_id=3844990](https://swinburne.instructure.com/courses/57020/pages/lect-ures-week-5?module_item_id=3844990) (accessed Apr. 28, 2024).
- [2]“Redirecting,” login.microsoftonline.com.  
[https://swinburne.instructure.com/courses/57020/pages/lab-oratory-week-6?module\\_item\\_id=3844994](https://swinburne.instructure.com/courses/57020/pages/lab-oratory-week-6?module_item_id=3844994) (accessed Apr. 28, 2024).
- [3]“What Is Full-Disk Encryption? – Definition from TechTarget.com,” WhatIs.  
[https://www.techtarget.com/whatis/definition/full-disk-encryption-FDE#:~:text=Full%2Ddisk%20encryption%20\(FDE\)%20is%20a%20security%20method%20for](https://www.techtarget.com/whatis/definition/full-disk-encryption-FDE#:~:text=Full%2Ddisk%20encryption%20(FDE)%20is%20a%20security%20method%20for)
- [4] A. Aarness, “EDR Security | What is Endpoint Detection and Response?,” CrowdStrike, Feb. 06, 2023.  
<https://www.crowdstrike.com/cybersecurity-101/endpoint-security/endpoint-detection-and-response-edr/>
- [5]“VPN Monitoring Methods | Junos OS | Juniper Networks,” www.juniper.net.  
<https://www.juniper.net/documentation/us/en/software/junos/vpn-ipsec/topics/topic-map/security-vpn-monitoring-methods.html> (accessed Apr. 28, 2024).
- [6] B. Team, “Mastering VPN Authentication: Best Practices for Enhanced Security,” blog.bio-key.com. <https://blog.bio-key.com/mastering-vpn-authentication-best-practices-for-enhanced-security>





