Objective: Deploy and operate a VPN server. Deploy a web server with pre-configured WordPress application. Examine stored passwords in the web server.

Discussion: OpenVPN is an open source VPN server that supports point-to-point and site-to-site connections. It allows remote users to connect to a local network as if their computer was directly connected to the local network. You will deploy an OpenVPN Docker container, retrieve the client configuration file and establish a VPN tunnel with the server. That VPN tunnel will allow you scan the internal Docker network and browse to a WordPress server in the Docker network.

Resources:

Openvpn client

Overview:

Deploy OpenVPN and WordPress servers
Connect to OpenVPN server
Test WordPress server
Examine and exploit stored passwords

Procedures:

Stop and delete all Docker containers and images. Verify the containers are stopped and the images are deleted.

We did this in a prior lab.

Install WordPress Server

The resources needed to deploy the OpenVPN and WordPress servers and the preconfigured network are stored in lab3.tar.gz. Download lab3.tar.gz and transfer it to your GCP instance. Extract the contents from the tarball. The extracted tarball has two directories. The docker compose file for WordPress starts the network for this lab and the WordPress server. We need to start WordPress first.

Change directories into /lab3. Use the docker-compose command to start WordPress (recall, we did this in this in Lab 1). Use the command **docker network ls** to list docker networks. You should see network lab3_DMZ-net.

Install OpenVPN

Next, we start the OpenVPN server. The directory /openvpn contains a Dockerfile, bash script openvpn.sh and server.conf. Start the OpenVPN server by executing the bash script. The script will build the OpenVPN image then start a container. A client configuration file (client.ovpn) will be created and downloaded in the process. Client.ovpn is needed to connect to the server.

Modify client.ovpn

By default, the OpenVPN server will become your computer's Default Gateway, meaning all traffic from your computer will be routed through the VPN when connected to the VPN. We want to avoid that. That configuration can be changed on either the server side or client side. We are going to modify client.ovpn.

We fix it by specifying preventing the client from pulling routes from the server and specifying our own route. Enter the following lines above the keys in client.ovpn.

route-nopull
route 172.16.0.0 255.255.0.0

See the example below. The modifications to the file are in red text.

```
client
dev tun
proto udp
route-nopull
route 172.16.0.0 255.255.0.0
resolv-retry infinite
nobind
persist-key
persist-tun
cipher AES-256-GCM
auth SHA512
verb 3
tls-client
tls-version-min 1.2
key-direction 1
remote-cert-tls server
;remote localhost 1194
;ca ca.crt
;cert client.crt
;key client.key
remote 35.225.124.82 1194
```

Connect to OpenVPN Server

The procedures used to open a VPN tunnel using OpenVPN depends on your host's operating system. Find an OpenVPN client installation guide for your OS. Install the OpenVPN client and using the client.ovpn configuration file you copied from the server, connect to the server. Note, OpenVPN uses UDP 1194. That port needs to be open on your GCP instance, so either add a rule to your firewall to open UDP 1194 or disable the firewall.

Nmap Scan

Now that you are connected to the VPN server, you can forward packets to the other guests in the VPN server's network through that tunnel. For example, we built a WordPress server. That server can be accessed through the VPN tunnel. Run a nmap scan against network 172.16.10.0/24 to discover the WordPress server. Hint: WordPress will be on TCP 80. Enumerate service version and write the output to a grepable file format. The following command writes the output to a file named lab3.txt in grepable

format. The hosts in the network 172.16.10.0/24 will be scanned for all ports and service versions will be enumerated on discovered ports.

```
nmap 172.16.10.0/24 -p- -sV -oG  lab3.nmap.txt
```

Now that you have the IP address you should be able to browse to that IP address and see the WordPress site. Note: your GCP server has very limited resources, the site could be very slow to load. Wait for it. You can verify that the WordPress serving is running by testing it with ping.

You should note the IP address for the WordPress server is a private, non-routable IP address. The only reason you were able to browse to that site is because the traffic was routed through the VPN tunnel. If you have problems browsing to the WordPress site, the place to start troubleshooting is with your VPN connection.

Capture a screenshot of your WordPress site displayed in a browser.

Stop and delete all Docker containers and images. Verify the containers are stopped and the images are deleted.

We did this in a prior lab.

Submission:

Nmap scan results in **lab3.nmap.txt**
Screenshot of the WordPress site