# Networking Training – Day-3 Hands-on Assignments

*Academy Batch 2025*

## Objectives

The goal of these assignments is not to teach you tools, but to *teach you the underlying networking concepts with it*. These assignments don’t focus on tools (hence the options in choosing a tool), rather how you understand and implement the concepts learnt with it. With that in mind, following are the objectives:

* Understand how load balancers distribute requests across multiple backends and explore the effect of different algorithms.
* Experience how CDNs improve latency and availability by comparing direct hosting with CDN-backed hosting.
* Explore how tunnels create private virtual networks over existing infrastructure.
* Show machine-to-machine communication and internet routing through a VPN.
* Expose a local service securely to the internet using a SaaS tunneling service.

## Assignment-1: Content Delivery Networks (CDNs)

Host a static page the “normal” way (on an EC2 Instance), then host the same page via a CDN service (Cloudflare, Netlify, Vercel, or any other). Compare the performance and behavior.

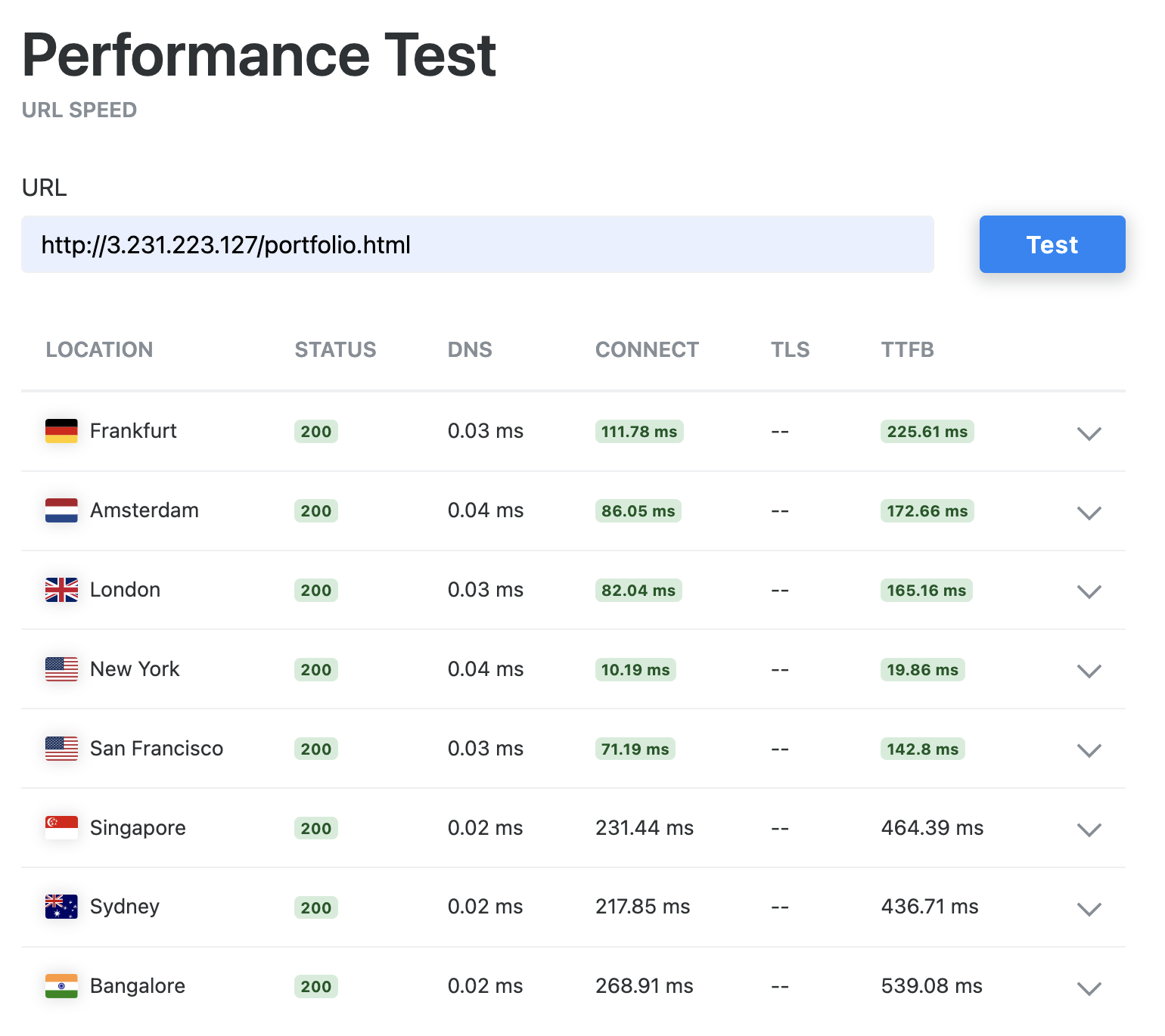
Ans :

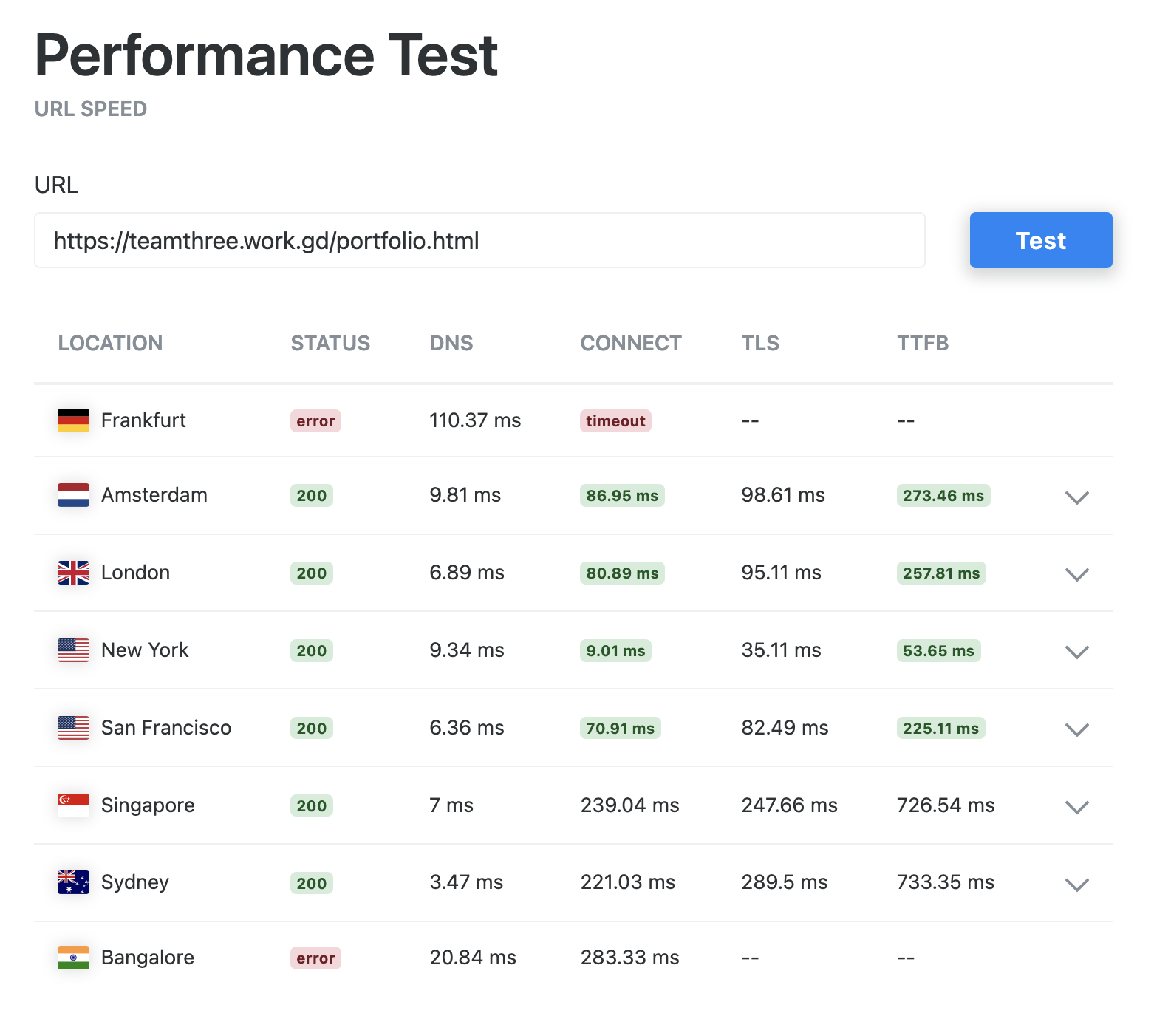
Hosted a webpage in EC2 Instance with SSL Certificate in public IP – 3.231.223.127 with domain – teamthree.work.gd

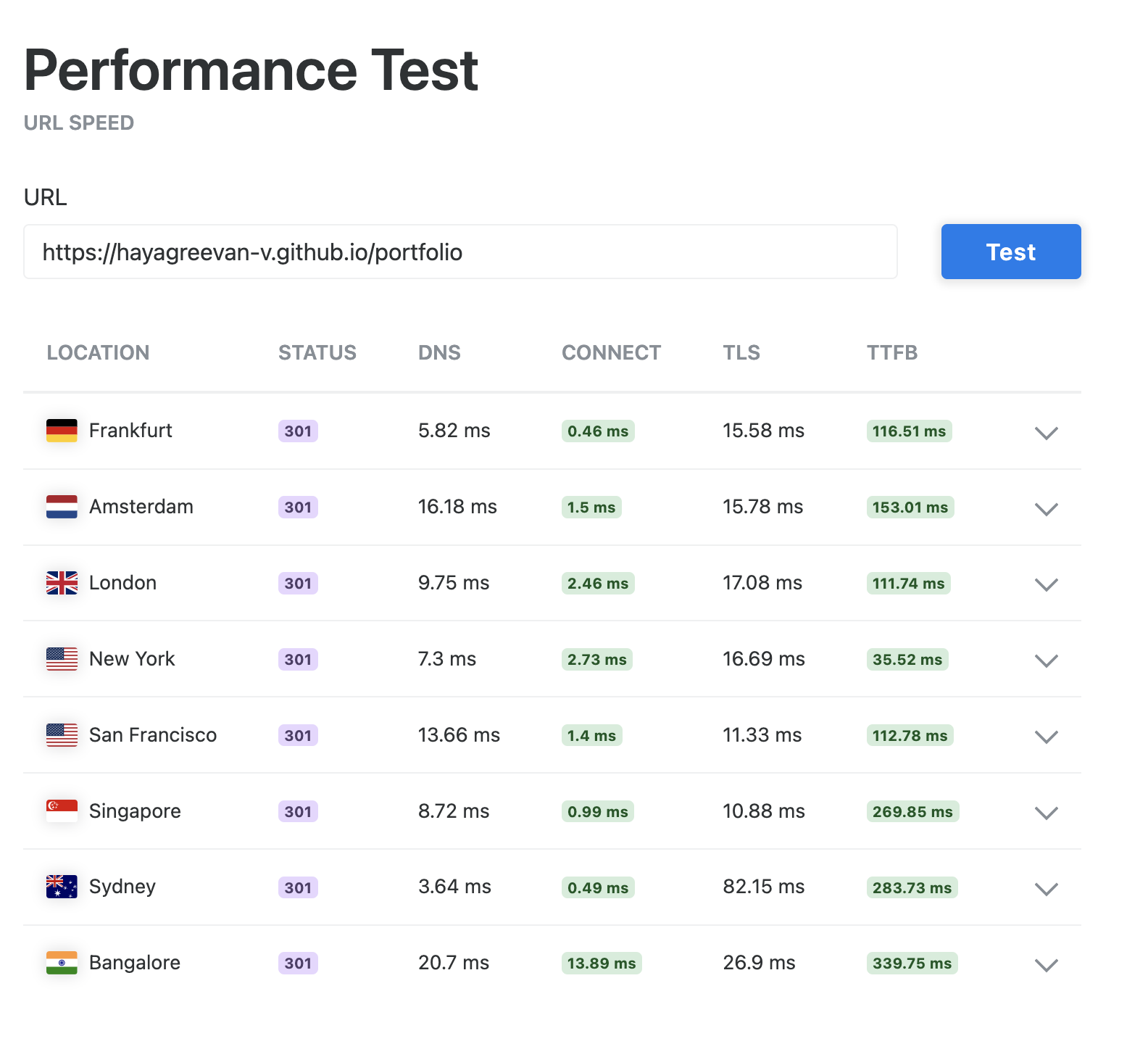
Hosted the same webpage in GitHub pages – hayagreevan-v.github.io/portfolio

Checked the performance of CDN using CDN Testing Website - <https://tools.keycdn.com/performance>

CDN webpage shows reduced latency and reduced Time-To-First-Byte (TTLB) as compared to static server hosted webpage.







## Assignment-2: Overlay Networks and Tunneling

1. Set-up a WireGuard VPN server on an AWS EC2 instance. Configure your machine to connect to it. Verify that your device has been assigned a private IP address from the VPN’s subnet. Check whether your location is masked.

**Server setup :**

Creating wireguard server private key :

* wg genkey > private

Creating wireguard server public key :

* wg pubkey < private > public

Setting up Wireguard network :

* ip link add dev wg0 type wireguard
* ip address add dev wg0 192.16.8.2.1/24
* wg set wg0 private-kay <server-private-key> peer <client-public-key> allowed-ips 192.168.2.2
* sudo ip link set up dev wg0
* sudo wg

Ennabling Firewall :

* sudo ufw allow <wg-port>/udp

Enabling IP Forwarding :

* sudo nano /etc/sysctl.conf

net.ipv4.ip\_forward=1

* sudo sysctl -p

net.ipv4.ip\_forward = 1

Setting up NAT :

* ip route | grep default
* sudo iptables -t nat -A POSTROUTING -o enX0 -j MASQUERADE

Allowing Forwarding between wg0 and enX0 :

* sudo iptables -A FORWARD -i wg0 -j ACCEPT
* sudo iptables -A FORWARD -o wg0 -j ACCEPT

Persist ip-table rules :

* sudo apt install iptables-persistent
* sudo netfilter-persistent save

Check status :

* sudo wg

**interface**: wg0

**public key**: rEeJGRu4LPn6xV4EIM3WT1lg56Z4JCQ/dQzVjo/fsH4=

**private key**: (hidden)

**listening port**: 44005

**peer**: y15zPrC527yqaSM/ZbOjuXsuSlqSSZwuq4C1eoTJRTo=

**endpoint**: <client-public-ip>:61665

**allowed ips**: 192.168.2.2/32

**latest handshake**: 1 minute, 8 seconds ago

**transfer**: 5.70 MiB received, 5.75 MiB sent

**In Local Device (Mac) :**

* brew install wireguard-tools
* cd /usr/local/etc/wireguard/

Creating wireguard client private key :

* wg genkey > private

Creating wireguard client public key :

* wg pubkey < private > public
* sudo nano /usr/local/etc/wireguard/wg0.conf

[Interface]

PrivateKey = <client-private-key>

Address = 192.168.2.2/24

DNS = 1.1.1.1

[Peer]

PublicKey = <server-public-key>

AllowedIPs = 0.0.0.0/0

Endpoint = 3.231.223.127:44005 // <server-public-ip>:<wg-port>

PersistentKeepalive = 25

Connect to VPN :

* wg-quick up wg0

Warning: `/usr/local/etc/wireguard/wg0.conf' is world accessible

[#] wireguard-go utun

[+] Interface for wg0 is utun4

[#] wg setconf utun4 /dev/fd/63

[#] ifconfig utun4 inet 192.168.2.2/24 192.168.2.2 alias

[#] ifconfig utun4 up

[#] route -q -n add -inet 0.0.0.0/1 -interface utun4

[#] route -q -n add -inet 128.0.0.0/1 -interface utun4

[#] route -q -n add -inet 3.231.223.127 -gateway 192.168.1.1

[#] networksetup -getdnsservers Thunderbolt Bridge

[#] networksetup -getsearchdomains Thunderbolt Bridge

[#] networksetup -getdnsservers Wi-Fi

[#] networksetup -getsearchdomains Wi-Fi

[#] networksetup -setdnsservers Wi-Fi 1.1.1.1

[#] networksetup -setsearchdomains Wi-Fi Empty

[#] networksetup -setdnsservers Thunderbolt Bridge 1.1.1.1

[#] networksetup -setsearchdomains Thunderbolt Bridge Empty

[+] Backgrounding route monitor

* Ifconfig

utun4: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1420

inet 192.168.2.2 --> 192.168.2.2 netmask 0xffffff00

Connect and Disconnect VPN :

* wg-quick up wg0
* wg-quick down wg0

1. In the training, we discussed local tunneling with SSH tunnels, where we were able to expose a service running remotely locally. Your task is to understand what remote tunneling is, find a scenario on your own, and implement remote tunneling with SSH tunnels.

Remote tunnelling enables us to access services hosted on the server from our local machine through Local Port Forwarding. Here we’re connecting to ssh from local machine, along with port forwarding from server to localhost

It also enables server to access the service available on local machine, we can set it up from local machine through Reverse Port Forwarding. Here we’re establishing ssh connection from local machine with port forwarding of a service which can be accessed by server as their localhost service.

These port forwarding will be available till the existance of ssh connection, once the connection is terminated, port forwarding is also terminated. It also supports Dynamic Port Forwarding.

* ssh -L 8080:3.231.223.127:80 -i Hayagreevan\_Vijayakumar.pem [ubuntu@3.231.223.127](mailto:ubuntu@3.231.223.127)

Able to interact with remote server port 80 from localhost:8080

Resources :

<https://www.digitalocean.com/community/tutorials/how-to-set-up-wireguard-on-ubuntu-20-04#step-3-creating-a-wireguard-server-configuration>

<https://youtu.be/O2mxQSqvsaM>

<https://www.wireguard.com/quickstart/>

https://stackoverflow.com/questions/5280827/can-someone-explain-ssh-tunnel-in-a-simple-way