# Private VPN Project

Full Documentation | 10-3-2023

Luke Tapanes

## Purpose

Virtual Private Networks are considered a critical component of security; thus, the purpose of this project is to gain a deeper understanding of the technology. In this project, I also want to increase my understanding and skills in networking, and why not own a private VPN in the process?

## Scope

This project will provide the VPN knowledge necessary to apply in the real-world. There are a few objectives that I would like to achieve in order to accomplish the overall goal of the project. The objectives are as follows:

- Select a Virtual Private Server (VPS) provider.
- Configure the VPS
- Update and patch the server.
- Configure and harden SSH.
- Install and configure OpenVPN.
- Transfer the configured VPN profile to the local machine.
- Run the VPN connection.
- Test the connection.

## Project

The first step in the project is to select what virtual private server (VPS) provider to use. I chose Vultr due to the fact that I haven't used this cloud service before, and the server is free upon signing up. I created an account, redeemed my free credit, and began configuring the components of the server. This can be seen in the screenshots below.

Products

Compute
Cloud Storage
Kubernetes
Container Registry
Databases
Load Balancers
Network
Orchestration

Products
Account
Support

## Choose Server

**Optimized Cloud Compute**
Virtual machines for more demanding business apps, e.g. production websites, CI/CD, video transcoding, or larger databases.
Starting from $28.00/mo — Dedicated vCPU

**Cloud Compute**
Virtual machines for apps with bursty performance, e.g. low traffic websites, blogs, CMS, dev/test environments, and small databases.
Starting from $2.50/mo — Shared vCPU

**Cloud GPU**
Virtual machines with fractional or full NVIDIA GPUs for AI, machine learning, HPC, visual computing and VDI. Also available as Bare Metal.
Starting from $21.50/mo — NVIDIA GPU + Dedicated vCPU

**Bare Metal**
Single tenant bare metal for apps with the most demanding performance or security requirements.
Starting from $120.00/mo — Physical CPU + Optional GPU

## CPU & Storage Technology

NEW **AMD High Performance**
Powered by latest generation AMD EPYC CPUs and NVMe SSD.

NEW **intel. High Performance**
Powered by new generations of Intel Xeon CPUs and NVMe SSD.

**intel. High Frequency**
Powered by 3GHz+ Intel Xeon CPUs and NVMe SSD.

**intel. Regular Performance**
Powered by previous generation Intel CPUs and regular SSD.

## Server Location

All Locations | America | Europe | Australia | Asia | Africa

| | |
|---|---|
| Miami | United States |
| Atlanta | United States |
| Chicago | United States |
| Dallas | United States |

Servers Qty: − 1 +
Summary: $14.40/month ($0.021/hour)
Deploy Now

Products

Compute
Cloud Storage
Kubernetes
Container Registry
Databases
Load Balancers
Network
Orchestration

| | |
|---|---|
| Miami | United States |
| Atlanta | United States |
| Chicago | United States |
| Dallas | United States |
| Honolulu | United States |
| Los Angeles | United States |
| Mexico City | Mexico |
| New York (NJ) | United States |
| Seattle | United States |
| Silicon Valley | United States |
| Toronto | Canada |
| London | United Kingdom |
| Amsterdam | Netherlands |
| Frankfurt | Germany |
| Madrid | Spain |
| Manchester | United Kingdom |
| Paris | France |
| Stockholm | Sweden |
| Warsaw | Poland |
| Tokyo | Japan |
| Bangalore | India |
| Delhi NCR | India |
| Mumbai | India |
| Osaka | Japan |
| Seoul | South Korea |
| Singapore | Singapore |
| Tel Aviv | Israel |
| Sydney | Australia |
| Melbourne | Australia |
| Johannesburg | South Africa |
| São Paulo | Brazil |
| Santiago | Chile |

## Server Image

Operating System | Marketplace Apps | Upload ISO | ISO Library | Backup | Snapshot

- AlmaLinux — Select Version
- Alpine Linux — Latest x64
- Arch Linux — Latest x64
- CentOS — Select Version
- Debian — 12 x64
- Fedora — Select Version
- Fedora CoreOS — Select Version
- Flatcar Container Linux — Select Version

Servers Qty: − 1 +
Summary: $14.40/month ($0.021/hour)
Deploy Now

**Debian**
12 x64

25 GB NVMe
$6/month
$0.009/hour

1 vCPU
1 GB Memory
2 TB Bandwidth

I set my location in London to test a geographically far distance. I also want to access content that is only accessible in the United Kingdom. Now that all of the components are configured, I clicked on “Deploy Now” and the machine was ready to be accessed.

| Server | OS | Location | Charges | Status |
|---|---|---|---|---|
| lukevpn<br>1024.00 MB AMD High Performance - 136.2 | | London | $0.01 | Running |

The first thing I did was SSH into the VPS and updated the packages and OS.

```
root@lukevpn:~# sudo apt update && sudo apt upgrade
```

I then created a new user and password then added the user to the sudoers list. This is to ensure best practices are met as using the root user to perform all tasks is dangerous.

```
luke@BocaServer: ~
root@lukevpn:~# useradd -m luket
root@lukevpn:~# passwd luket
New password:
BAD PASSWORD: The password is shorter than 8 characters
New password:
Retype new password:
passwd: password updated successfully
root@lukevpn:~#
```

Now that a new user has been configured, I then began hardening the system be securing SSH. The three main security principles here are to disable root login, disable password authentication, and enable public key authentication. To do this I first generated the keys on the local machine which can be seen in the screenshot below.

```
luke@BocaServer: ~/.ssh
luke@BocaServer:~$ cd .ssh
luke@BocaServer:~/.ssh$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/luke/.ssh/id_rsa): lukevps_key
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in lukevps_key
Your public key has been saved in lukevps_key.pub
The key fingerprint is:
SHA256:pSOecK6l1NbbwSMtpIlMeBjF774zhcmCgzBHs9Bo6cE luke@BocaServer
The key's randomart image is:
+---[RSA 4096]----+
|.o...            |
|oE+..            |
|oooo .    .      |
|o.o+  .  o       |
|.+o.+oooS        |
|. o+.B=B.+       |
|   .+oX.+ =      |
|   . =+  = o     |
|    o .+. .      |
+----[SHA256]-----+
luke@BocaServer:~/.ssh$
```

I then copy the public key over to the VPS and store it in the /.ssh directory.

```
luke@BocaServer:~/.ssh$ scp lukevps_key.pub luket@136.[redacted]:
luket@136.2[redacted] password:
lukevps_key.pub                                   100%  741     7.3KB/s   00:00
```

I then went back to the VPS. From here, I edited the /etc/ssh/sshd_config file to disable root login, disable password authentication, and enable public key authentication. This can be seen in the screenshots below.

```
luke@BocaServer: ~
  GNU nano 7.2                    /etc/ssh/sshd_config *

PermitRootLogin no


# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.
```

```
PubkeyAuthentication yes
```

```
PasswordAuthentication no
```

Next I restarted the sshd process to make sure the setting updated. To do this, I issued the following command:

```
luket@lukevpn:~$ sudo systemctl restart sshd
```

I then exited the VPS and tested to see if I could successfully login to the server with just the key. The attempt was successful, and the syntax used can be seen in the screenshot below.

```
luke@BocaServer:~$ ssh -i .ssh/lukevps_key luket@136.2[redacted]
```

Now that SSH has been successfully configured, I moved on to configuring the VPN. First, I installed OpenVPN.

```
luket@lukevpn:~$ sudo apt install openvpn
```

To help out with the configuration process, I installed a script from GitHub. This step isn't necessary; however, it speeds up the configuration process. The syntax can be seen in the screenshot below.

```
luket@lukevpn:~$ wget https://raw.githubusercontent.com/angristan/openvpn-instal
l/master/openvpn-install.sh
--2023-10-03 15:10:40--  https://raw.githubusercontent.com/angristan/openvpn-ins
tall/master/openvpn-install.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 2606:50c0:800
3::154, 2606:50c0:8000::154, 2606:50c0:8001::154, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|2606:50c0:80
03::154|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 40820 (40K) [text/plain]
Saving to: ‘openvpn-install.sh’

openvpn-install.sh  100%[===================>]  39.86K  --.-KB/s    in 0s

2023-10-03 15:10:40 (101 MB/s) - ‘openvpn-install.sh’ saved [40820/40820]
```

I then ran the script which walked me through the VPN configuration. Now the VPN has officially been configured and named desktop.ovpn.

```
luket@lukevpn:~$ ls
desktop.ovpn  lukevps_key.pub  openvpn-install.sh
```

The next step now is to get this file on the local machine. To do this, I will use Secure File Transfer Protocol (SFTP). This can be seen in the screenshot below.

```
luke@BocaServer:~$ sftp -i .ssh/lukevps_key luket@136.2[redacted]
Connected to 136.2[redacted]
sftp> ls
desktop.ovpn        lukevps_key.pub     openvpn-install.sh
sftp> get desktop.ovpn
desktop.ovpn        lukevps_key.pub     openvpn-install.sh
sftp> get desktop.ovpn
Fetching /home/luket/desktop.ovpn to desktop.ovpn
desktop.ovpn                                  100% 2772    13.7KB/s   00:00
sftp> exit'
Unterminated quoted argument
sftp> exit
luke@BocaServer:~$ ls
 BocaCloud      Documents   Pictures   Templates  'VirtualBox VMs'
 Desktop        Downloads   Public     Test
 desktop.ovpn   Music       snap       Videos
luke@BocaServer:~$
```

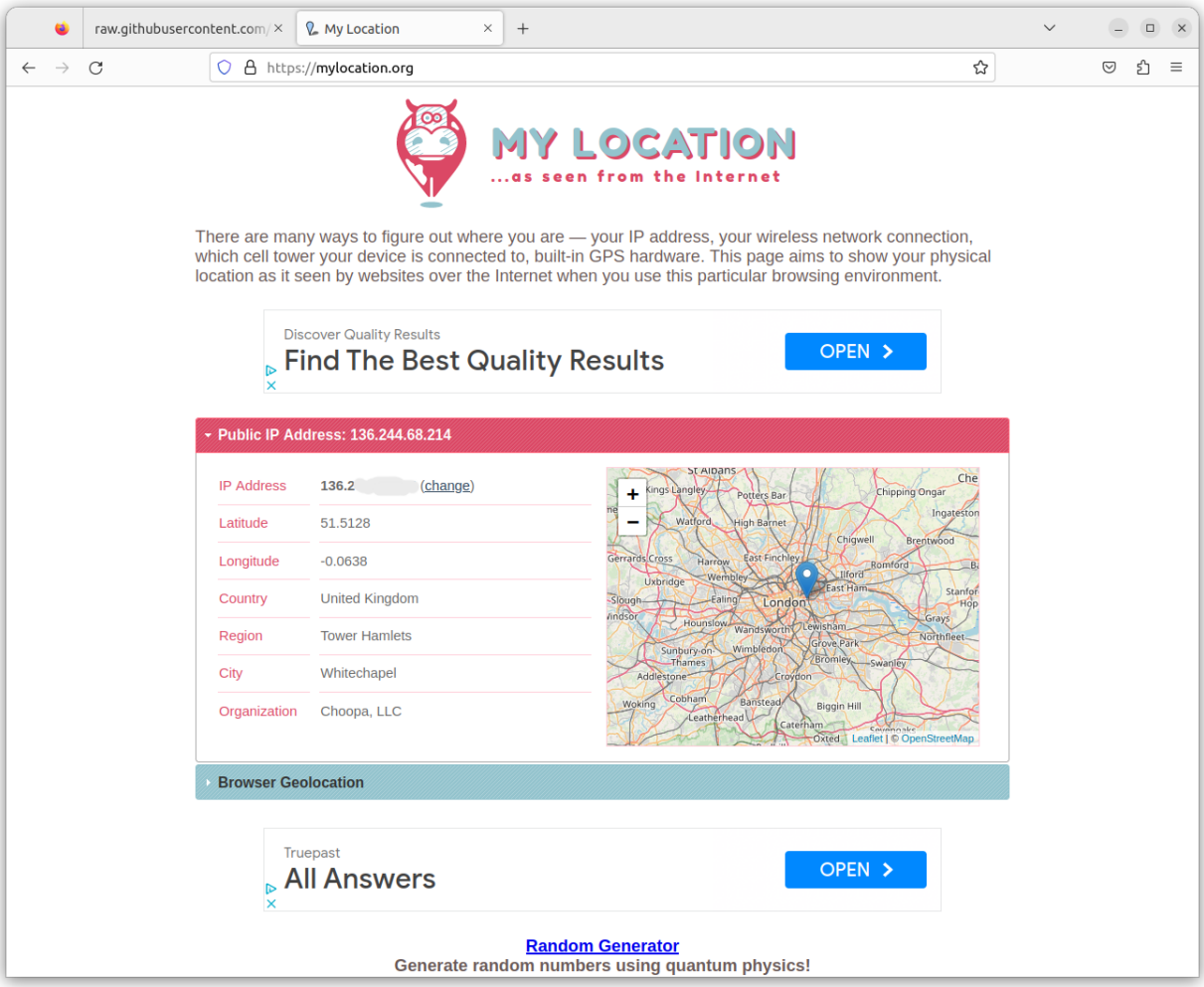
Now comes the moment of truth. It is time to test the VPN to see if it works. To do this, I issue the command “openvpn.”

```
luke@BocaServer:~$ sudo openvpn desktop.ovpn
```

Sure enough, it worked! To make sure that it is working, I check the IP address which I now have a different one.

```
tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST>  mtu 1500
        inet 10.8.0.2  netmask 255.255.255.0  destination 10.8.0.2
        inet6 fe80::4a00:d69b:7fa0:b18c  prefixlen 64  scopeid 0x20<link>
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 500
```

Just to double check, I did a location lookup and sure enough, my location is in the middle of London!




## Lessons Learned

This was a fairly simple project but also very instructive. I learned a lot more about VPN technology and how it works at a fundamental level. I also learned more about networking concepts in the process. Overall, this was a successful project and I achieved all of the objectives that I set out to do. To recap, in this project I:

- Selected a Virtual Private Server (VPS) provider.
- Configured a VPS
- Updated and patched the server.
- Configured and hardened SSH.
- Installed and configured OpenVPN.
- Transferred the configured VPN profile to the local machine.
- Ran the VPN connection.
- Tested the connection.

Here are a few resources that inspired and helped me out with this project.

https://www.youtube.com/watch?v=gxpX_mubz2A&t=851s

https://www.youtube.com/watch?v=Lk_v6Q0YsNo