ORACLE

Deploying Distributed Multi-Node Solution for Roving Edge Devices with Compute Cloud@Customer

Version 1.1

Public

ORACLE

Purpose statement

This whitepaper looks at a multi-node setup using Oracle's Roving Edge Devices (REDs) and Compute Cloud@Customer. Acting as the central hub, Compute Cloud@Customer connects securely with REDs in different locations through VPN technology, making data sharing smooth and reliable. This solution helps industries like oil and gas, healthcare, and others bring Oracle's cloud services to remote or hard-to-reach-areas.

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

ORACLE

Table of contents

Introduction

In today's world, secure access to systems from remote locations is essential. Oracle's Compute Cloud@Customer and Roving Edge devices (REDs) make it easier to bring cloud services to remote areas. This is especially useful for industries like Oil and Gas, Healthcare, Manufacturing, and Logistics, where real-time data and reliable operations are critical. Setting up OpenVPN on Compute Cloud@Customer as a central server allows businesses to securely connect Roving Edge devices in different locations. This ensures safe data transfer, supports important operations, and keeps teams connected, even in areas with limited resources.

This paper covers the benefits, setup, and steps for using OpenVPN on Compute Cloud@Customer with Roving Edge. It's a solution that protects data, improves efficiency, and keeps businesses ahead.

Note: This content is provided for informational purposes and self-supported guidance only. Consultancy or other assistance related to the content is not covered under the Oracle Support contract or associated service requests. If you have questions or additional needs, then please reach out to your Oracle Sales contact directly.

Overview of OpenVPN Access Server

OpenVPN Access Server is a powerful, secure, and flexible VPN solution that enables remote users to connect safely to private networks. It features an intuitive web-based interface for managing connections, configuring VPN settings, and monitoring network activity with ease. By encrypting traffic, it ensures secure communication and protects sensitive data from potential threats. OpenVPN Access Server is highly scalable, making it suitable for businesses of all sizes, from small teams to large enterprises. Designed for simplicity, it streamlines deployment, management, and maintenance, allowing users to set up and maintain VPN connections effortlessly.

Prerequisites

The environment assumes the provisioning of the following resources:

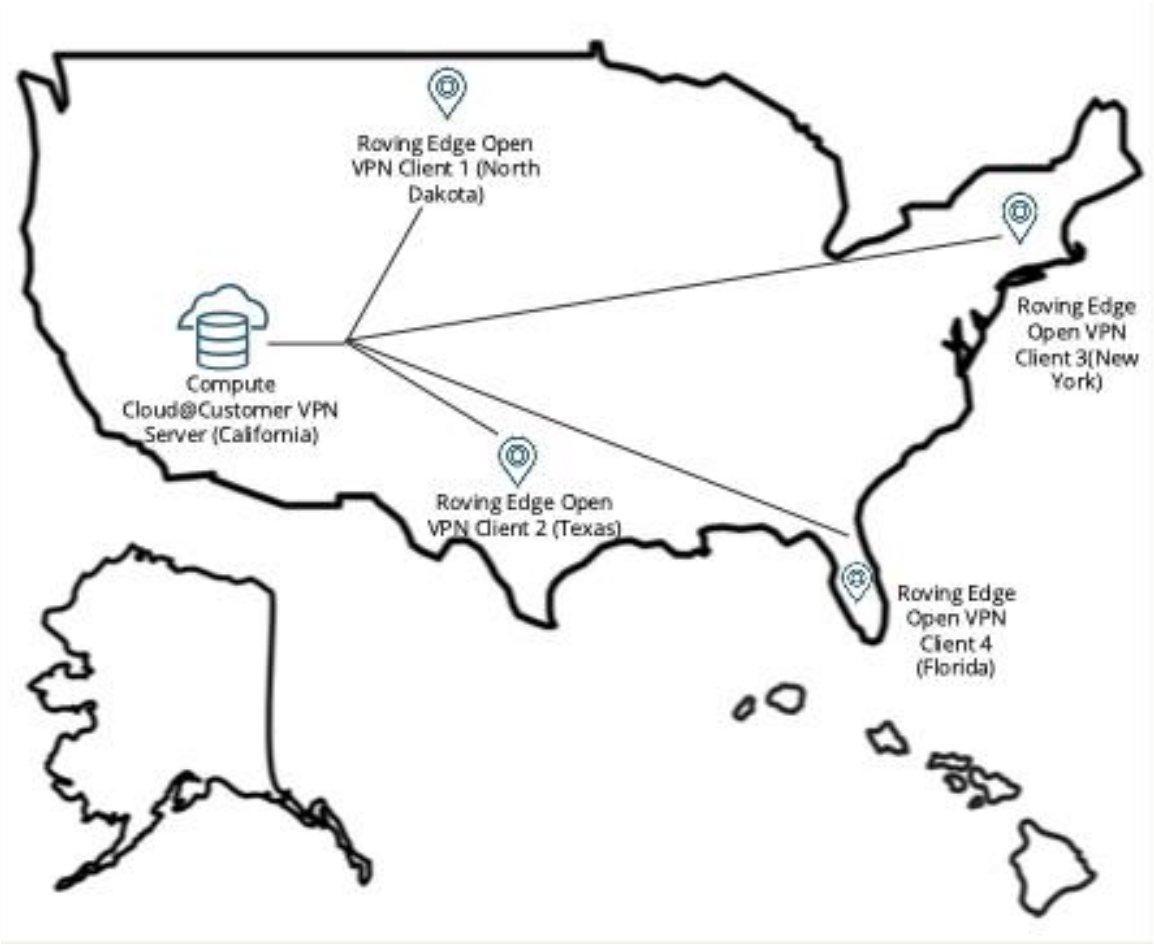
- **Access to Oracle Compute Cloud@Customer:** Ensure you have the necessary access credentials to Oracle's Compute Cloud@Customer infrastructure, including administrative permissions for setting up and managing virtual machines.
- **Access to Oracle Roving Edge Infrastructure:** Ensure you have the necessary access credentials to Oracle's Roving Edge Infrastructure, including administrative permissions for deploying and managing virtual machines and resources within the Roving Edge devices.
- **OpenVPN Access Server License/Software:** A valid OpenVPN Access Server license is required for advanced features.
- **SSH Key Pair for Server Access:** An SSH key pair to securely connect to the Oracle Compute Cloud@Customer and Roving Edge instances.
- **Virtual Cloud Network (VCN)**: Create a VCN with a DNS label. Enable DNS hostnames by selecting "Use DNS hostnames in this VCN."
- **IAM Policies**: Set up Identity and Access Management (IAM) policies to allow user and resource access in the compartment.
- **Subnets**: Ensure at least one public and one private subnet exist in the VCN. These will handle secure network traffic.

Architecture Overview

The setup for using OpenVPN with Oracle's Roving Edge devices ensures secure, encrypted communication between Virtual Cloud Networks (VCNs) and Virtual Machines (instances) on different Roving Edge devices. It includes an OpenVPN server or gateway and multiple clients, each in its own VCN, allowing secure connections across remote locations.

A map of the U.S. shows how Roving Edge devices are spread out and connected to Compute Cloud@Customer. While Compute Cloud@Customer is typically used as the OpenVPN server, a Roving Edge device can also act as the server if needed, giving flexibility for edge deployments. Each Roving Edge device works as a VPN client and a gateway for other devices on its network that don't have OpenVPN installed.

This setup provides secure and reliable connectivity for distributed operations, making it easier to manage networks in remote locations.



Key Components:

1. **OpenVPN Gateway/Server:**

   NOTE: The OpenVPN Server will serve as both a server for the Roving Edge (OpenVPN Clients) and as a gateway for LAN clients (without OpenVPN installed). This dual role enables routing to specific clients or all clients within the private VCN behind the OpenVPN server, allowing seamless connectivity between them.

   - The OpenVPN Gateway/Server operates on a Linux-based instance in the Compute Cloud@Customer region (California). It serves as the central hub for encrypted traffic.
   - The VPN interface for the gateway uses an IP address within the 10.8.0.0/24 range, specifically 10.8.0.1 in this configuration.
   - It is configured to manage secure connections to and from multiple client instances across geographically dispersed locations.

2. **Roving Edge OpenVPN Clients**:

- Each instance, located within its own VCN, acts as an OpenVPN client and gateway for the LAN clients running within the same VCN/private subnetwork (without OpenVPN client installed.). These instances use OpenVPN to establish secure tunnels to the OpenVPN Gateway/Server.
- Each client connects back to the OpenVPN Gateway, ensuring secure communication between all edge nodes.
- The architecture depicts the following clients:
    1. **Client 1 (North Dakota):** IP `10.8.0.2`
    2. **Client 2 (Texas):** IP `10.8.0.3`
    3. **Client 3 (New York):** IP `10.8.0.4`
    4. **Client 4 (Florida):** IP `10.8.0.5`

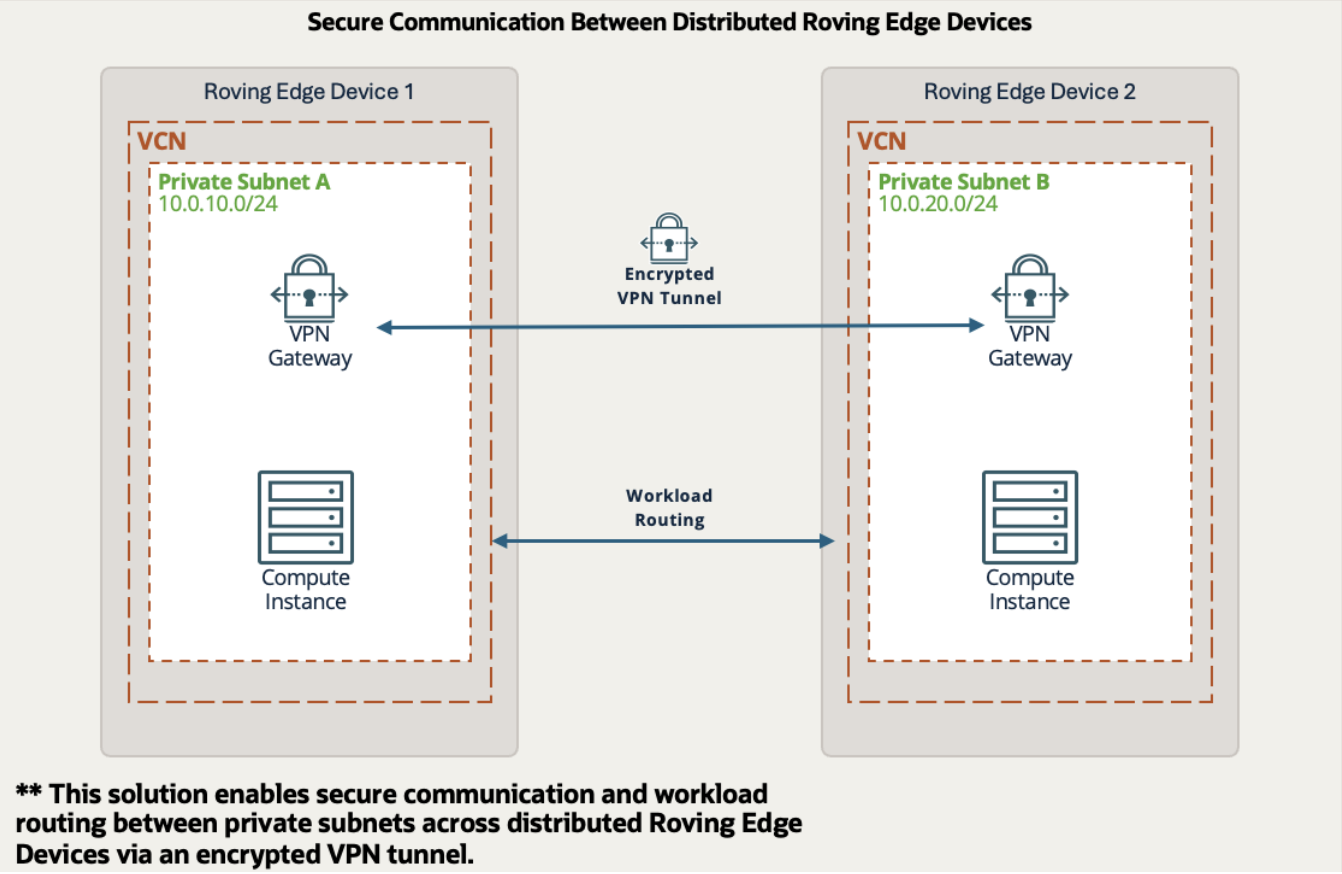
3. **VPN Interfaces and Secure Network Traffic:**
    - Each instance has its own VPN interface to handle encrypted traffic between the instance and the OpenVPN server or gateway. This keeps all data secure and protected from outside threats.
    - The VPN interfaces use IP addresses from the 10.8.0.x subnet, allowing the devices to communicate smoothly within the network.

Network Flow:

The architecture enables secure connectivity by routing all traffic through the OpenVPN Gateway. Each instance establishes an encrypted tunnel to the OpenVPN Gateway, ensuring that data transmitted between the Instances and the edge devices remains private and protected.

This architecture provides a scalable, secure solution for edge computing environments, making it suitable for various use cases such as field deployments and remote data aggregation, all while maintaining the confidentiality of transmitted data.

Instances Configuration

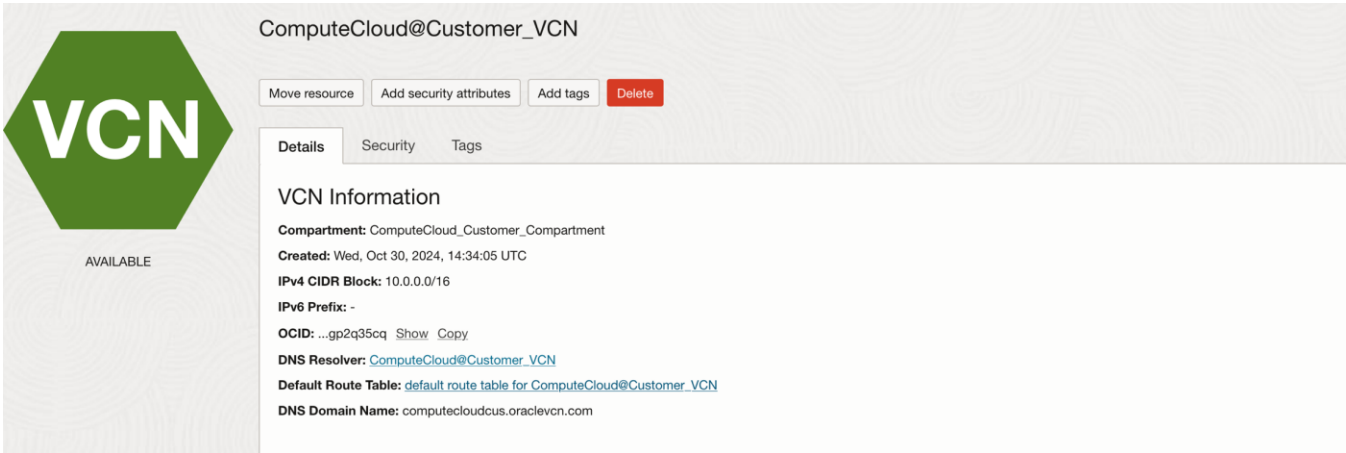


Oracle Linux on Compute Cloud@Customer

Create a Virtual Cloud Network (VCN)

Why it's important: The VCN provides a secure, isolated network environment for the OpenVPN server to communicate with other resources. Public and private subnets separate external-facing components from internal resources for added security.

- VCN Name: ComputeCloud@Customer_VCN
- CIDR Block: 10.0.1.0/24
- Enable DNS resolution and DNS hostnames



Configure Security Lists:

Why it's important: Security lists manage ingress and egress traffic. You need to allow OpenVPN traffic (UDP 1194) and HTTP/HTTPS for management (TCP 443).

- Allow UDP on port 1194 (OpenVPN)
- Allow TCP on port 443 for web access
- Configure additional rules based on network needs

Ingress Rules

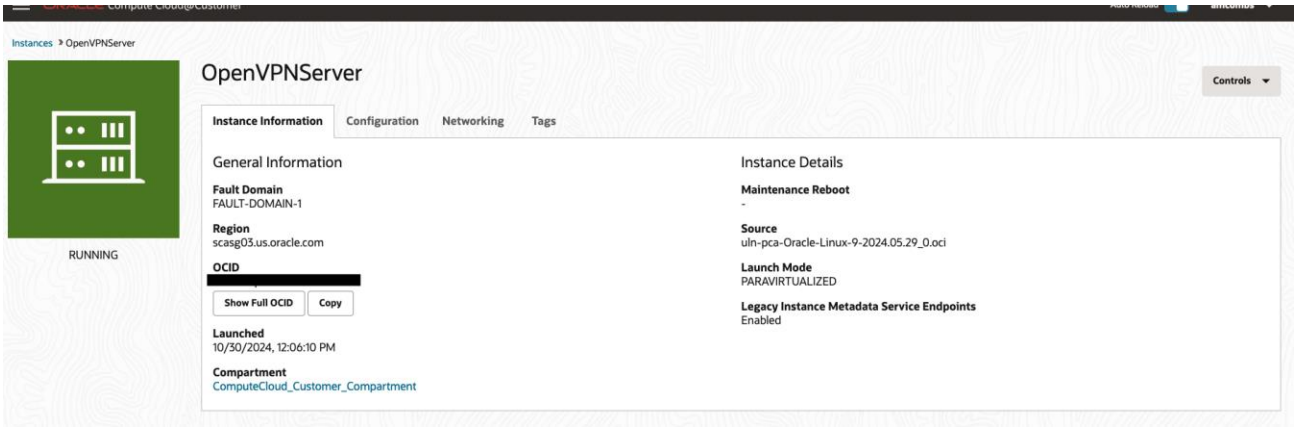
Add Ingress Rules | Edit | Remove

Stateless	Source	IP Protocol	Source Port Range	Destination Port Range	Type and Code	Allows	Description
No	0.0.0.0/0	TCP	All	22		TCP traffic for ports: 22 SSH Remote Login Protocol	
No	0.0.0.0/0	ICMP			3, 4	ICMP traffic for: 3, 4 Destination Unreachable: Fragmentation Needed and Don't Fragment was Set	
No	10.0.0.0/16	ICMP			3	ICMP traffic for: 3 Destination Unreachable	
No	10.0.3.0/24	UDP	All	1194		UDP traffic for ports: 1194	Allow OpenVPN traffic from Roving Edge VCNs.
No	10.0.4.0/24	UDP	All	1194		UDP traffic for ports: 1194	Allow OpenVPN traffic from Roving Edge VCNs.
No	10.0.5.0/24	UDP	All	1194		UDP traffic for ports: 1194	Allow OpenVPN traffic from Roving Edge VCNs.
No	0.0.0.0/0	TCP	All	443		TCP traffic for ports: 443 HTTPS	Allow web-based access for management.

0 selected Showing 7 items 1 of 1

Launch the Instance:

- Choose Oracle Linux 8/9, with a public subnet for OpenVPN access
- Assign SSH keys for remote management



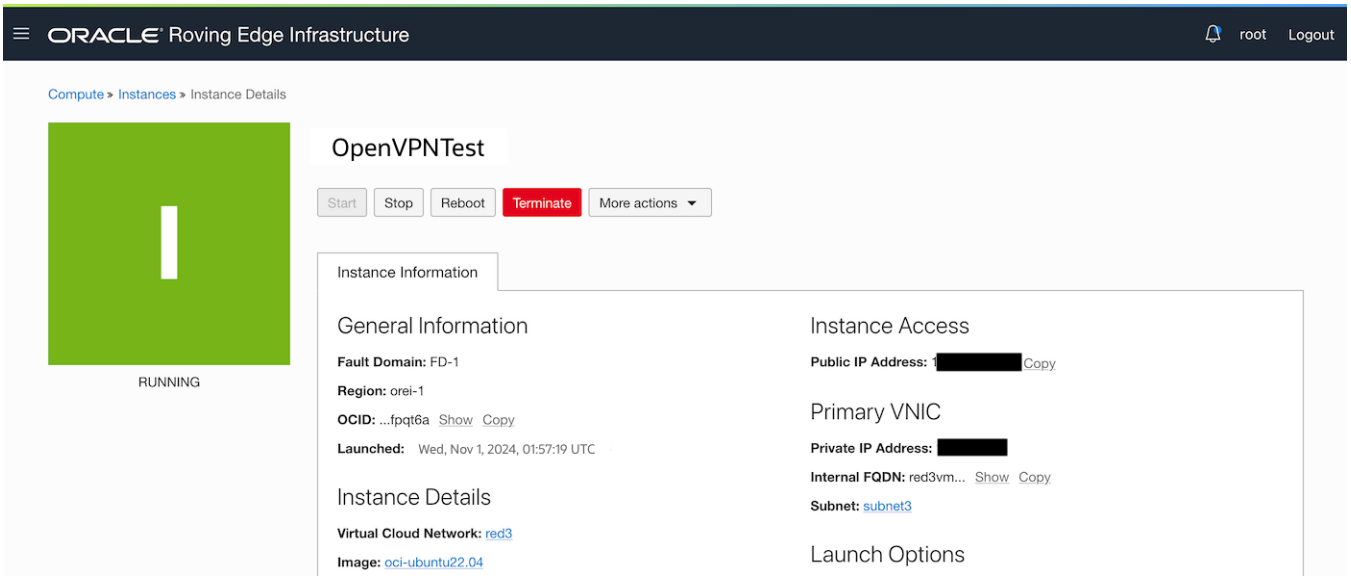
Oracle Linux on Roving Edge:

Network and Security Setup

- Each Roving Edge device uses its own VCN (e.g., VCN 10.0.3.0/24)
- Configure private subnets and static routes to ensure inter-VCN communication

Instance Launch

- Setup each OpenVPN client on a Linux Instance within its VCN
- Assign private Ips and configure VPN interface according to each site's specific network settings (e.g., New York: 10.0.5.0)



OpenVPN Configuration

Oracle Linux

To install OpenVPN on Oracle Linux, follow these steps. These instructions assume you're using Oracle Linux 8 or 9, which is based on the same package management system as Red Hat-based distributions.

Step 1: Update the System

First, make sure that your system is up to date by running the following command:

sudo dnf update -y

Step 2: Install EPEL Repository

EPEL (Extra Packages for Enterprise Linux) provides OpenVPN packages. You can install it using the following commands:

For Oracle Linux 8:

sudo dnf install epel-release -y

For Oracle Linux 9:

sudo dnf install oracle-epel-release-el9 -y

Step 3: Install OpenVPN

Once the EPEL repository is installed, you can install OpenVPN by running:

sudo dnf install openvpn -y

Step 4: Generate Server and Client Certificates

OpenVPN requires certificates for secure communication. To generate the necessary certificates, you'll need to install easy-rsa:

sudo dnf install easy-rsa -y

Then create a directory for the certificates:

mkdir ~/openvpn-ca
cd ~/openvpn-ca
cp -r /usr/share/easy-rsa/3/ ./*

Next, initialize the Public Key Infrastructure (PKI):

./easyrsa init-pki

To build a Certificate Authority (CA):

./easyrsa build-ca

Step 5: Create Server Certificate and Keys

To create a server certificate and key:

./easyrsa gen-req server nopass
./easyrsa sign-req server server

Step 6: Configure OpenVPN

Copy the server certificate and key files to OpenVPN's directory:

sudo cp ~/openvpn-ca/pki/private/server.key /etc/openvpn/
sudo cp ~/openvpn-ca/pki/issued/server.crt /etc/openvpn/
sudo cp ~/openvpn-ca/pki/ca.crt /etc/openvpn/

Create the server configuration file:

vi /etc/openvpn/server.conf

An example configuration might look like this:

ORACLE

Red Hat Linux

Red Hat Repository Configuration

Step 1: Ensure the Base Repositories Are Enabled

First, check if your RHEL system has the required repositories enabled:

1. **Check subscription**: If you're using RHEL, ensure that your system is properly subscribed to Red Hat's repositories. Run the following command:

sudo subscription-manager status

If you aren't subscribed, you can register the system:

sudo subscription-manager register --username <your-Red-Hat-account-username> --password <your-password>

Then, attach a subscription:

sudo subscription-manager attach --auto

Enable RHEL repositories:

sudo subscription-manager repos --enable rhel-9-for-x86_64-baseos-rpms

sudo subscription-manager repos --enable rhel-9-for-x86_64-appstream-rpms

Step 2: Reinstall the Repositories

If the base repositories are not installed or configured correctly, you can reinstall them by running:

sudo dnf install redhat-release -y

sudo dnf repolist

Step 1: Update your system

Before installing OpenVPN, update your system packages:

sudo dnf update -y

Step 2: Download and Install the EPEL RPM Manually

If dnf can't find the package automatically, you can manually download and install the EPEL repository package from the Fedora project site.

1. **Download the EPEL RPM for RHEL 9:** Use dnf to download and install the EPEL repository from the Fedora project's official site.

sudo dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm

Verify Installation: After installing the EPEL repository, you can verify that the EPEL repository is enabled by running:

sudo dnf repolist

You can now install packages from the EPEL repository.

To install OpenVPN on RHEL 9.3, follow these steps:

Install OpenVPN

Install OpenVPN using the following command:

sudo dnf install openvpn -y

Configuring OpenVPN

1. **Install Easy-RSA.** For the OpenVPN server, you'll need Easy-RSA to generate encryption keys and certificates. To install, run the following command line in your OpenVPN Server (not on clients):

   sudo dnf install easy-rsa -y

2. **Generate Keys and Certificates (For Server Setup)**
   - Use Easy-RSA to create a Certificate Authority (CA), server certificate, and client certificates. Find the Easy-RSA's script to be utilized to create the new Certificate Authority (CA), server certificate, and client certificates.
   - Run the command line **/usr/share/easy-rsa/3.1.6/easyrsa init-pki** to initialize the Public Key Infrastructure (PKI) directory. Below is the output of this command:

```
Notice
------
'init-pki' complete; you may now create a CA or requests.

Your newly created PKI dir is:
* /usr/share/easy-rsa/pki
Using Easy-RSA configuration:
* /usr/share/easy-rsa/pki/vars

IMPORTANT:
  Easy-RSA 'vars' template file has been created in your new PKI.
  Edit this 'vars' file to customise the settings for your PKI.
  To use a global vars file, use global option --vars=<FILE>
```

3. **Next steps are to build the CA for the OpenVPN server. Run the following command line below:**

   /usr/share/easy-rsa/3.1.6/easyrsa build-ca

```
Using Easy-RSA 'vars' configuration:
* /usr/share/easy-rsa/pki/vars

Using SSL:
* openssl OpenSSL 3.0.7 1 Nov 2022 (Library: OpenSSL 3.0.7 1 Nov 2022)

Enter New CA Key Passphrase: Enter your new passphrase

Confirm New CA Key Passphrase: Confirm your new passphrase
```

```
..+.....+...+....+...+..+.+..........+....+..........+.....++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++*.....+.++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++*...+....+..........+...+.........+..............+............+......+...+..+...+.
+..............+.................+.+.....+.+...+..+.........+.+.................+.....+.+..
...............+...+.............+..+.+......+...+...+..+...+.........+.....+...+......+.+..
.........+..+....+.........+...+..+...+...+.................+......+.......+..+......+...+.
+...+......+.........+..+.+..+...+....+............+++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++
......+.+...........+.............+..+......+.+......+...+.....++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++*...+.+...........++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++*.......+...+..+..............+......+........+.+..........+.+.....+..
.+.+..........+.........+.+...+.........+..+......+.......+.....+.+.....+.........+.+......+...
.....+...+.........+...+.......+.........+..+.........+...+.+.........+......+..+.......+.......
....+..........+......+.....+......+....+.....+......+.......+......+.....+....+.....+.........
.....+..............+..........+..+.+........+................+...+...+.........+...........+..
.....+...+.....+.................+.+......+...+.....+....+...+........+.........+.+.....+......
+..............+.+...+..+.........+...+......+.........+......+..............+...+...+..+....
...+......+..+......+.+...........+...+.+...+......+..+.........+.+............+.....+.........
+......+..........+...+.+...........+...+..+....+.....+......+.........+.+...........+..+....
......+...+..........+.+......+..+...+.......+...+........+.+.....+....+...........+.........+
...++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Common Name (eg: your user, host, or server name) [Easy-RSA CA]: Enter the name of your server

Notice
------
CA creation complete. Your new CA certificate is at:
* /usr/share/easy-rsa/pki/ca.crt
```

You new ca.crt for your OpenVPN server has been successfully created and available on the following folder:
/usr/share/easy-rsa/pki/ca.crt

4. **Generate the Server Certificate and Key:** To generate the OpenVPN server certificate and key, run the following command line below:

```
/usr/share/easy-rsa/3.1.6/easyrsa gen-req server nopass
```

NOTE: This command creates a server key and a certificate signing request (CSR). You will be prompted to enter a Common Name (e.g., server).

```
Using Easy-RSA 'vars' configuration:
* /usr/share/easy-rsa/pki/vars

Using SSL:
* openssl OpenSSL 3.0.7 1 Nov 2022 (Library: OpenSSL 3.0.7 1 Nov 2022)
...+..+.............+........+.....+.+........+..........+...+.......+...+........+......
..+....++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*..........+..........
....+....+...++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.+..........+..
+.........+........+.....+.....+.+..+...........+.+...+..+.......+.....+.....+...+......+.
..............+...+.+...+.....+..........+.+...+..+....+................+........+........
.+...+.........+.....+.......+.....+........+.....+..+.+.....+.........+.....+.+.....+.....
.+.......+.............+...+..+...+........+......+....................+....+.....+......
...+...........+...+...........+....+............+...+....+...+................+...+......
+......+........+........+...+........+......+...+.......+.....+.......+........+.......+..+..
.+.........+......+............+............+..+...+......+...+.........+...+..........
...+.+..+...+...........+...+....+.....+.....+........+....+.....+........+..+........+.+.
.+.........+..+....+............+............+..+...+....+.....+.+..+...+......+.+.......
..+......+..+......+.+............+.+.....+....+.....++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++
......+...+......+.....+....+........+................+.......+..+...........+.+..+......+.+.
............++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.........+......
.....+.........+..+....+..........+...+.+...+......+........+.....+......+......+....+..+....
+......+........+....+..+.........++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++*.......+...+.....+.+.....+.........+......+.....+.........+...........+..+....+.........
....+......+........+......+.+...+..+.........+........+...+........+...+...+.......+..+....+
.............+...+.......+..........+...........+....+.....+................+..........+.
..+..............+...........+...+..........+......+...........+.+.......................
....+......+......+.+.....+....+...+...+.....+...........+...+....+...++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++
-----


-----
Common Name (eg: your user, host, or server name) [server]:ovpn-41

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

5. **Next, sign the server certificate with the CA:**Type yes when prompted to confirm the signing. This creates the server certificate (server.crt).

```
/usr/share/easy-rsa/3.1.6/easyrsa sign-req server server
```

```
/usr/share/easy-rsa/3.1.6/easyrsa sign-req server server
Using Easy-RSA 'vars' configuration:
* /usr/share/easy-rsa/pki/vars

Using SSL:
* openssl OpenSSL 3.0.7 1 Nov 2022 (Library: OpenSSL 3.0.7 1 Nov 2022)
You are about to sign the following certificate:
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.
Request subject, to be signed as a server certificate
for '825' days:

subject=
    commonName                = ovpn-41

Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes

Using configuration from /usr/share/easy-rsa/pki/openssl-easyrsa.cnf
Enter pass phrase for /usr/share/easy-rsa/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'ovpn-41'
Certificate is to be certified until Feb  4 17:47:34 2027 GMT (825 days)

Write out database with 1 new entries
Data Base Updated

Notice
------
Certificate created at:
* /usr/share/easy-rsa/pki/issued/server.crt
```

6. Generate the Diffie-Hellman Parameters

Diffie-Hellman (DH) parameters are used to ensure secure communication between the VPN clients and the server. Generate the DH parameters with the following command:

```
/usr/share/easy-rsa/3.1.6/easyrsa gen-dh

Using Easy-RSA 'vars' configuration:
* /usr/share/easy-rsa/pki/vars

Using SSL:
* openssl OpenSSL 3.0.7 1 Nov 2022 (Library: OpenSSL 3.0.7 1 Nov 2022)
Generating DH parameters, 2048 bit long safe prime
```

```
..+...........+.....................................................................
.....................................................+...............................
.....................................................+...............................
.....................................................+..........+.........+..........
................................................................................
.........+.....................................................................+.....
................................................................................
................................................................................
.....................................................................................+..
.........................+.....................................+.....................
.....................................................++*++*++*++*++*++*++*++*++*++*++*++*++*
++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*
++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*
DH parameters appear to be ok.

Notice
------

DH parameters of size 2048 created at:
* /usr/share/easy-rsa/pki/dh.pem
```

7. **Generate a Shared TLS Auth Key:** This step is optional but recommended to provide an extra layer of security. Generate a shared TLS authentication key (also known as an HMAC key):

```
openvpn --genkey secret /etc/openvpn/ta.key
```

8. **Generate Client Certificates and Keys:** On the OpenVPN server, generate a certificate and key pair for each OpenVPN client. Run the following command to create a client certificate request:

   NOTE: Replace client1 with a unique name for each client (e.g., client1, client2, etc.).

For client 1, run the following command line:

```
/usr/share/easy-rsa/3.1.6/easyrsa gen-req client1 nopass
```

```
/usr/share/easy-rsa/3.1.6/easyrsa gen-req client1 nopass
Using Easy-RSA 'vars' configuration:
* /usr/share/easy-rsa/pki/vars

Using SSL:
* openssl OpenSSL 3.0.7 1 Nov 2022 (Library: OpenSSL 3.0.7 1 Nov 2022)
```

```
..+....+.....+.+........+.+...+......+.........++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++*....+.+..+....+.....+.........+............+...++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++*.+...............+.............+.+..............+.........+
.....+.+..................+..........+...+...........+....+.........+........+...+.+...+.....+
.......+.....+...+.+..................+.......+...+...+..+..........+...+......+...........+...+.
+........+.+.........+...+........+.+........+.+.....+....+..+.+..+..........+..+.+........+....
+......+...........+.........+.........+..+.......+..+.........+.......+............+..+.+.....
+....+......+...+.........+..+.+..+...............+.+..+...+.+...+......+......+......+..+......
.+...........+....+...+..+.+.....+.........+......+............+.........+..+............+....
.......+.........+.............+.+..+....+.....+............+..+......+.........+......+.....
............+.+......+...+.....+.+...+..+.+..............+............+...+..+.......+......+.
.......+.........+..+.......+..+...+.+......+...+........+.........+.+..+.+.....+.+...........
..+.........+...+...+....+..+............+......+................+............+.+......+...+..
....+.....+............+....+...+................+......+.......+...+..+.+..+....+...+..+...+..
.+...+.......+..+..........+......+.....+.+......+.......................+....+......+..........
.+.+.....+....+.....+...+...+..........+.....+...+.+...+..+...+.+............+........++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++
.......+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*..+......+.+.....+....+
.....+......+.+...+..+...+....++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*
......+...+....+......+.........+...........+..........+..+...+....+...........+..........+..+..
..........+...+..........+...+................+.......+......+.....+.........+...+...+.........
.+......+........+..........+..+.+............+..+.........+....+..+.+..+.+.....+....+....+....
.+..........+..+.........+....+..............+...+....+...+...+..+......+....+..+......+.......+
........+..............+....+...+...........+.+.....+...+............+.+...+......+...+..+.+...
......++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Common Name (eg: your user, host, or server name) [client1]: Enter the name of your OpenVPN
client

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

9. **Once the CSR is generated, sign it with the CA. You will be prompted to type yes to confirm the signing:**

   For client1:

   cd /usr/share/easy-rsa

```
/usr/share/easy-rsa/3.1.6/easyrsa sign-req client client1
Using Easy-RSA 'vars' configuration:
* /usr/share/easy-rsa/pki/vars

Using SSL:
* openssl OpenSSL 3.0.7 1 Nov 2022 (Library: OpenSSL 3.0.7 1 Nov 2022)
You are about to sign the following certificate:
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.
Request subject, to be signed as a client certificate
for '825' days:

subject=
    commonName                = client1

Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes

Using configuration from /usr/share/easy-rsa/pki/openssl-easyrsa.cnf
Enter pass phrase for /usr/share/easy-rsa/pki/private/ca.key:
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 Feb  4 18:11:07 2027 GMT (825 days)

Write out database with 1 new entries
Data Base Updated

Notice
------
Certificate created at:
* /usr/share/easy-rsa/pki/issued/client1.crt
```

For client2:

```
/usr/share/easy-rsa/3.1.6/easyrsa sign-req client client2
Using Easy-RSA 'vars' configuration:
* /usr/share/easy-rsa/pki/vars

Using SSL:
* openssl OpenSSL 3.0.7 1 Nov 2022 (Library: OpenSSL 3.0.7 1 Nov 2022)
You are about to sign the following certificate:
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.
Request subject, to be signed as a client certificate
for '825' days:
```

```
subject=
    commonName                = client2

Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes

Using configuration from /usr/share/easy-rsa/pki/openssl-easyrsa.cnf
Enter pass phrase for /usr/share/easy-rsa/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'client2'
Certificate is to be certified until Feb  4 18:11:16 2027 GMT (825 days)

Write out database with 1 new entries
Data Base Updated

Notice
------
Certificate created at:
* /usr/share/easy-rsa/pki/issued/client2.crt
```

10. **Copy Certificates and Keys to the OpenVPN Directory:** After generating the server and client certificates, keys, and other necessary files, copy them to the OpenVPN directory. For the server:

```
cp /usr/share/easy-rsa/pki/ca.crt /etc/openvpn/
cp /usr/share/easy-rsa/pki/issued/server.crt /etc/openvpn/
cp /usr/share/easy-rsa/pki/private/server.key /etc/openvpn/
cp /usr/share/easy-rsa/pki/dh.pem /etc/openvpn/
```

For the clients, you need to copy the following files to the client's /etc/openvpn directory securely. Copy the files listed below for each client. Example:

```
/usr/share/easy-rsa/pki/ca.crt
pki/issued/client1.crt
pki/private/client1.key
ta.key (if using TLS authentication)
```

11. **Configure OpenVPN:** You'll need to configure OpenVPN by creating or modifying a configuration file. Copy the example server configuration file to the OpenVPN directory. Example:

    sudo cp /usr/share/doc/openvpn/sample/sample-config-files/server.conf /etc/openvpn/server.conf

    Modify the OpenVPN configuration. Starting with the OpenVPN server:

    vi /etc/openvpn/server/server.conf

IMPORTANT: We need to configure OpenVPN infrastructure to allow routing for the OpenShift instances running behind the OpenVPN servers and clients' subnets. For this, ensure you have the correct IP routes for all your subnets listed on the OpenVPN server.conf file using the "push" configuration option, the correct subnets listed on the "route" configuration option in your OpenVPN server. Also, you need to create a file under /etc/openvpn/client for each OpenVPN client you have, then add the "iroute" configuration option. Iroute configuration option is used in the client-specific configuration file (CCD) to indicate which client can route traffic for a given subnet. It helps the OpenVPN server forward packets correctly when clients are acting as gateways for specific subnets. The iroute directive ensures that the server routes traffic destined for a specific subnet through the appropriate client.

```
# OpenVPN server configuration
port 1194                    # The port OpenVPN will listen on (default is 1194)
proto udp                    # Protocol: UDP is generally preferred over TCP
dev tun                      # Use TUN for routed mode

# Server-side certificate and key files
ca /etc/openvpn/ca.crt       # Certificate Authority (CA) file
cert /etc/openvpn/server.crt       # Server certificate file
key /etc/openvpn/server.key # Server private key file
dh /etc/openvpn/dh.pem      # Diffie-Hellman parameters (used for key exchange)
tls-auth /etc/openvpn/ta.key 0

# Network settings
server 10.8.0.0 255.255.255.0 # VPN subnet for OpenVPN clients
client-config-dir /etc/openvpn/client

# Push routes to the client to redirect their traffic through the VPN
push "route 10.8.0.0 255.255.255.0"
push "route 10.0.3.0 255.255.255.0"
push "route 10.0.4.0 255.255.255.0"
push "route 10.0.5.0 255.255.255.0"

# Configures the server's routing table by adding a route to a specific subnet or network
route 10.0.4.0 255.255.255.0
route 10.0.5.0 255.255.255.0

# Enable client-to-client communication
client-to-client

# Ensure IP forwarding is enabled
persist-key
persist-tun

# Keepalive settings (to detect dropped clients)
keepalive 10 120
```

```
# Security settings
data-ciphers AES-256-GCM:AES-128-GCM
data-ciphers-fallback AES-256-CBC
auth SHA256                     # Authentication method

# Enable log file and verbosity
verb 3
log-append /var/log/openvpn.log

# Allow OpenVPN to run as a background process
daemon
```

Create a new file named client1 under /etc/openvpn/client/ folder, then add the iroute option as listed below. In our example, our OpenVPN client1 is running on the 10.0.4.0/24 subnet, so we need to add the iroute option for the 10.0.4.0/24 on client1 file:

more /etc/openvpn/client/client1

iroute 10.0.4.0 255.255.255.0

The same configuration needs to be done for the client2 or any other client you might have. In our example, our OpenVPN client2 is running on the 10.0.5.0/24 subnet, so we need to add the iroute option for the 10.0.5.0/24 on client2 file as following:

more /etc/openvpn/client/client2

iroute 10.0.5.0 255.255.255.0

Configure Client

For the client, you need to create a configuration file for the clients. Since we are working with two OpenVPN clients (client1 and client2), both files need to be created. Below is the recommended configuration for the OpenVPN clients:

On OpenVPN client1 use the following configuration file:

```
# OpenVPN client configuration
client                        # Indicates that this is a client configuration
dev tun                       # Use TUN for routed mode
proto udp                     # Protocol (must match the server's protocol)
remote 10.145.142.8 1194     # Server's IP address and port
```

```
# Client-side certificate and key files

ca /etc/openvpn/ca.crt      # Certificate Authority (CA) file
cert /etc/openvpn/client1.crt      # Client certificate file
key /etc/openvpn/client1.key       # Client private key file
tls-auth /etc/openvpn/ta.key 1 # Pre-shared TLS key (optional, must match server)



# Network configuration
remote-cert-tls server         # Ensure that we're connecting to a valid server

# Security settings
data-ciphers AES-256-GCM:AES-128-GCM
data-ciphers-fallback AES-256-CBC

#cipher AES-256-CBC             # Encryption cipher (must match the server)
auth SHA256                    # Authentication method (must match the server)

# Verbosity level (3 is default, increase to debug)
verb 3

# Resolv retry if the connection fails (useful for dynamic IPs)
resolv-retry infinite

# Prevent multiple connections from the same client
nobind

# Persist keys and tunnel across restarts
persist-key
persist-tun
```

On OpenVPN client2 use the following configuration file:

```
# OpenVPN client configuration
client                        # Indicates that this is a client configuration
dev tun                       # Use TUN for routed mode
proto udp                     # Protocol (must match the server's protocol)
remote 10.145.142.8 1194     # Server's IP address and port

# Client-side certificate and key files

ca /etc/openvpn/ca.crt  # Certificate Authority (CA) file
cert /etc/openvpn/client2.crt   # Client certificate file
key /etc/openvpn/client2.key    # Client private key file
tls-auth /etc/openvpn/ta.key 1 # Pre-shared TLS key (optional, must match server)
```

```
# Network configuration
remote-cert-tls server         # Ensure that we're connecting to a valid server

# Security settings
data-ciphers AES-256-GCM:AES-128-GCM
data-ciphers-fallback AES-256-CBC

#cipher AES-256-CBC             # Encryption cipher (must match the server)
auth SHA256                    # Authentication method (must match the server)

# Verbosity level (3 is default, increase to debug)
verb 3

# Resolv retry if the connection fails (useful for dynamic IPs)
resolv-retry infinite

# Prevent multiple connections from the same client
nobind

# Persist keys and tunnel across restarts
persist-key
persist-tun
```

Enable IP Forwarding on OpenVPN server and OpenVPN clients.

To allow traffic to be forwarded between OpenVPN clients and OpenShift instances across different subnets, enable IP forwarding by modifying /etc/sysctl.conf on both, OpenVPN server and OpenVPN clients:

vi /etc/sysctl.conf

Add the following line:

net.ipv4.ip_forward = 1

Apply the changes:

sudo sysctl -p

NOTE: If you have firewall enabled on the OpenVPN hosts, adjust firewall rules to the following listed below:

If you are running a firewall (e.g., firewalld), allow OpenVPN traffic through the firewall:

sudo firewall-cmd --add-service=openvpn --permanent

sudo firewall-cmd –reload

Start and Enable OpenVPN Service

To start and enable OpenVPN and ensure it starts on boot, use the following commands:

On OpenVPN server: cp /etc/openvpn/server.conf to /etc/openvpn/server/

sudo systemctl start openvpn-server@server.service

sudo systemctl enable openvpn-server@server.service

Check the status of the service with:

sudo systemctl status openvpn-server@server.service

On OpenVPN clients: cp /etc/openvpn/client1.conf to /etc/openvpn/server/

sudo systemctl start openvpn-client@client1.service

sudo systemctl enable openvpn-client@client1.service

Check the status of the service with:

sudo systemctl status openvpn-client@client1.service

Additional Resources

- Oracle Compute Cloud@Customer
- Oracle Compute Cloud@Customer documentation
- Roving Edge Infrastructure
- Oracle Linux Configuring Virtual Private Networks

ORACLE

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle

Author: Anderson Souza