

How to Expose Services using Vultr Load Balancer and VPC 2.0

Learn how to securely expose your services to the internet using Vultr Load Balancer with VPC 2.0. Step-by-step guide for optimal network configuration.

Contents

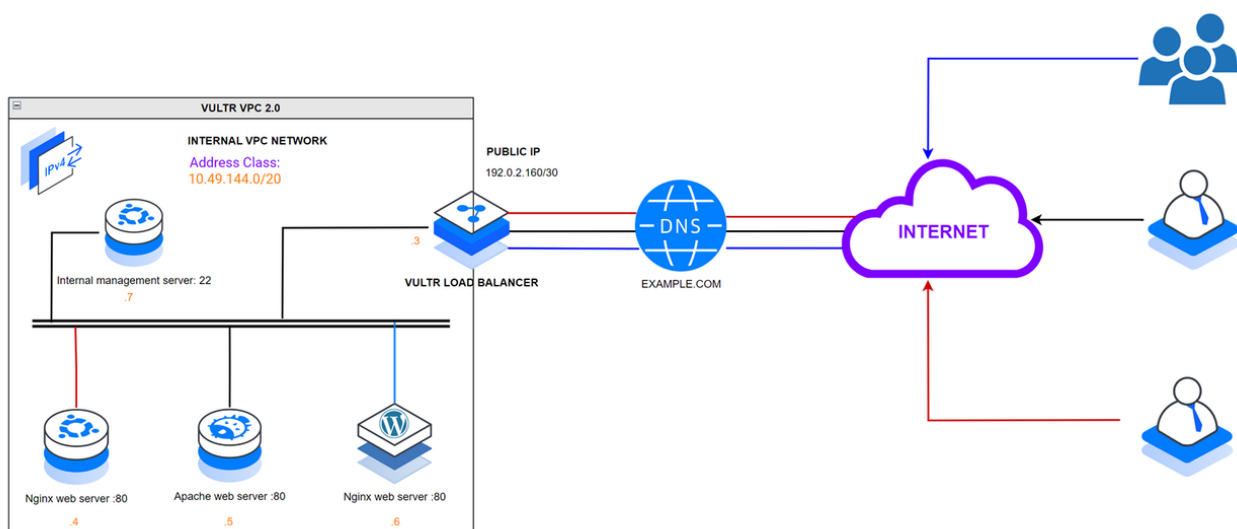
01	Introduction	3
02	Prerequisites	5
03	Attach VPC 2.0 Hosts to the Vultr Load Balancer	5
04	Configure the VPC 2.0 Host Management Server	6
05	Set Up the Internal VPC 2.0 Host Services	9
06	Set Up the Vultr Load Balancer Forwarding Rules to Expose VPC 2.0 Host Services	20
07	Test Access to the VPC 2.0 Host Services	21
08	Best Practices	23
09	Conclusion	24

Introduction

Vultr Load Balancer is a network service that distributes traffic across multiple servers connected to optimize the performance, reliability, and availability of hosted applications or services. A Vultr Load Balancer connects directly to all target host servers using a backend network interface. In a VPC 2.0 network, a Vultr load balancer securely exposes internal Vultr VPC2.0 network hosts by forwarding and distributing specific external network traffic to the internal VPC host services.

This article explains how to expose services using a Vultr Load Balancer in a VPC 2.0 network. You will set up multiple servers in a Vultr VPC 2.0 network, run similar services on each server, and securely expose all hosts using a Vultr Load Balancer.

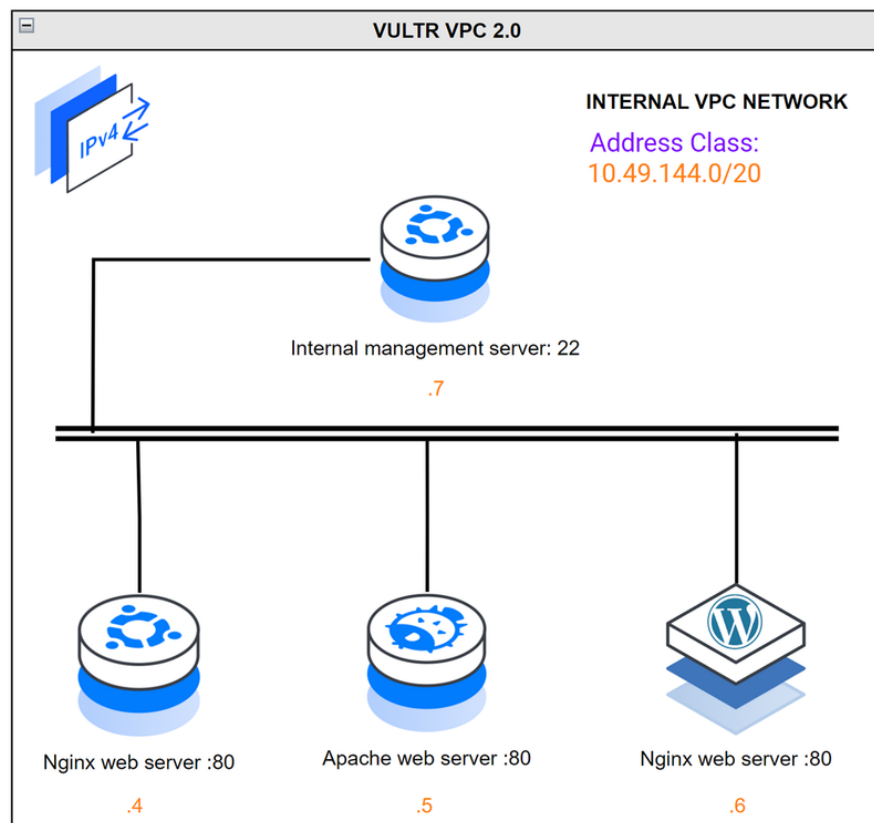
Example VPC2.0 Network Topology



Within the VPC 2.0 network:

- **Vultr Load Balancer:** Works as the default gateway to all internal VPC 2.0 host services. All external requests to the public IP address `192.0.2.160` or the associated domain `example.com` translate to the respective VPC 2.0 services.

- **Host 1:** Runs the main internal web server using Nginx and accepts incoming connections on the HTTP port `80`.
- **Host 2:** Works as a backup web server using Apache.
- **Host 3:** Works as a backup web server using Nginx.
- **Host 4:** Works as the internal VPC 2.0 host management workstation. A public network interface is available on the server with the IP address `192.0.2.100` and connects to all internal hosts over SSH using the VPC 2.0 interface.



Internal VPC 2.0 host network addresses:

- **Vultr Load Balancer:** `10.49.144.3`
- **Host 1:** `10.49.144.4`
- **Host 2:** `10.49.144.5`
- **Host 3:** `10.49.144.6`
- **Host 4:** `10.49.144.7`

Prerequisites

In a single Vultr location:

- Deploy a [Vultr Load Balancer](#) and attach it to a Vultr VPC 2.0 network.
- Create a new [domain A record](#) that points to the Vultr Load Balancer public IP address.
- Deploy a [Ubuntu 20.04 server](#) to use as the management workstation `host 4`.
- Deploy a [Ubuntu instance](#) to use as the `host 1` Nginx web server.
- Deploy a [Rocky Linux instance](#) to use as the `host 2` Apache web server.
- Deploy a [Debian server](#) to use as the `host 3` Nginx web server.

Note

Add all instances to the same Vultr VPC 2.0 network.

Attach VPC 2.0 Hosts to the Vultr Load Balancer

1. Open the [Vultr Customer Portal](#).
2. Navigate to **Products** and click **Load Balancers** on the main navigation menu.
3. Click your target load balancer to open the instance control panel.
4. Click **Add Instance** within the **Linked Instances** within the **Overview Tab**.

Overview Configuration Metrics

Healthy Instances 0/0

SSL Redirect Disabled

Current Charges \$0.02

Add Instance

1024.00 MB AMD High Performance - 13...

+ Attach Instance

Linked Instances + Add instance

Name	Status
No instances attached to this loadbalancer	

5. Select your target VPC host from the list within the **Add Instance** pop-up. For example, `Host 1` and click **Attach instance** to add the instance to your load balancer.
6. Repeat the process for all VPC 2.0 hosts `host 2` and `host 3` to attach them to the load balancer.
7. Verify that all instances are attached to the load balancer and listed in the **Linked Instances** section.

Linked Instances + Add instance

Name	Status
Host-2 1024.00 MB AMD High Performance -	Running
Host-1 1024.00 MB AMD High Performance -	Running
Host-3 4096.00 MB Regular Cloud Compute -	Running

Configure the VPC 2.0 Host Management Server

Configure the VPC 2.0 management server to set up access to all internal hosts that run similar services to expose with the Vultr Load Balancer. Based on the example network topology, the `host 4` management server is not attached to the load balancer, has a public network interface, and accepts outgoing SSH connections to manage other internal hosts.

1. Access the Ubuntu internal VPC 2.0 management server using SSH.

CONSOLE

```
$ ssh root@192.0.2.100
```

2. Create a new standard user account with sudo privileges. For example, `vpcadmin`.

CONSOLE

```
# adduser vpcadmin && adduser vpcadmin sudo
```

3. Switch to the user.

CONSOLE

```
# su vpcadmin
```

4. View the server network interfaces and verify that the VPC 2.0 network interface is active with a private IP address.

CONSOLE

```
$ ip a
```

Output:

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 56:00:04:d5:77:b6 brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.100/23 metric 100 brd 192.0.2.255 scope global dynamic enp1s0
        valid_lft 85764sec preferred_lft 85764sec
    inet6 2a05:f480:3000:2321:5400:4ff:fed5:77b6/64 scope global dynamic
```

```
mngtmpaddr noprefixroute
    valid_lft 2591659sec preferred_lft 604459sec
    inet6 fe80::5400:4ff:fed5:77b6/64 scope link
        valid_lft forever preferred_lft forever
3: enp8s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc mq state UP group
default qlen 1000
    link/ether 5a:01:04:d5:77:b6 brd ff:ff:ff:ff:ff:ff
    inet 10.49.144.7/20 brd 10.49.159.255 scope global enp8s0
        valid_lft forever preferred_lft forever
    inet6 fe80::5801:4ff:fed5:77b6/64 scope link
        valid_lft forever preferred_lft forever
```

5. Test access to the Vultr Load Balancer internal IP using the Ping utility.

CONSOLE

```
$ ping 10.49.144.3
```

Output:

```
PING 10.49.144.3 (10.49.144.3) 56(84) bytes of data.
64 bytes from 10.49.144.3: icmp_seq=1 ttl=62 time=211 ms
64 bytes from 10.49.144.3: icmp_seq=2 ttl=62 time=0.868 ms
64 bytes from 10.49.144.3: icmp_seq=3 ttl=62 time=0.806 ms

--- 10.49.144.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 0.806/71.053/211.487/99.301 ms
```

6. Allow SSH access on the server using the default UFW firewall.

CONSOLE

```
$ sudo ufw allow ssh
```

7. Block all incoming network connections on the server.

CONSOLE

```
$ sudo ufw default deny incoming
```

Output:

```
Default incoming policy changed to 'deny'  
(be sure to update your rules accordingly)
```

8. Reload the UFW table rules to apply the firewall changes.

CONSOLE

```
$ sudo ufw reload
```

The VPC 2.0 management server acts as the main gateway to the internal network hosts over SSH while the Vultr Load Balancer acts as the external gateway depending on the exposed ports. Follow the additional sections below to use the management server to establish SSH connections, install services, and apply configuration changes on each of the target internal hosts.

Set Up the Internal VPC 2.0 Host Services

Host 1 (Main Web Server)

1. Access the Ubuntu server using SSH.

CONSOLE

```
$ ssh root@10.49.144.4
```

2. Create a non-root user with sudo privileges.

CONSOLE

```
# sudo adduser sysadmin && adduser sysadmin sudo
```

3. Switch to the new user.

```
CONSOLE
```

```
# su - sysadmin
```

4. View the server network interfaces and note the public network interface name.

```
CONSOLE
```

```
$ ip a
```

Output:

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 56:00:04:d5:77:b6 brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.101/23 metric 100 brd 192.0.2.255 scope global dynamic enp1s0
        valid_lft 85764sec preferred_lft 85764sec
    inet6 2a05:f480:3000:2321:5400:4ff:fed5:77b6/64 scope global dynamic mngtmpaddr noprefixroute
        valid_lft 2591659sec preferred_lft 604459sec
    inet6 fe80::5400:4ff:fed5:77b6/64 scope link
        valid_lft forever preferred_lft forever
3: enp8s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc mq state UP group default qlen 1000
    link/ether 5a:01:04:d5:77:b6 brd ff:ff:ff:ff:ff:ff
    inet 10.49.144.4/20 brd 10.49.159.255 scope global enp8s0
        valid_lft forever preferred_lft forever
    inet6 fe80::5801:4ff:fed5:77b6/64 scope link
        valid_lft forever preferred_lft forever
```

5. Test access to the Vultr Load Balancer VPC address using the Ping utility.

```
CONSOLE
```

```
$ ping 10.49.144.3
```

Output:

```
PING 10.49.144.3 (10.49.144.3) 56(84) bytes of data.  
64 bytes from 10.49.144.3: icmp_seq=1 ttl=62 time=211 ms  
64 bytes from 10.49.144.3: icmp_seq=2 ttl=62 time=0.868 ms  
64 bytes from 10.49.144.3: icmp_seq=3 ttl=62 time=0.806 ms  
  
--- 10.49.144.3 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2002ms  
rtt min/avg/max/mdev = 0.806/71.053/211.487/99.301 ms
```

6. Update the server packages.

```
CONSOLE
```

```
$ sudo apt update
```

7. Install the Nginx web server application.

```
CONSOLE
```

```
$ sudo apt install nginx -y
```

8. Verify the Nginx system service status.

```
CONSOLE
```

```
$ sudo systemctl status nginx
```

Output:

```
● nginx.service - A high performance web server and a reverse proxy server  
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
```



```
Status: active
```

To	Action	From
--	-----	----
22/tcp	ALLOW	Anywhere
80	ALLOW	10.49.144.0/20
22/tcp (v6)	ALLOW	Anywhere (v6)

12. Navigate to the Nginx web root directory `/var/www/html`.

```
CONSOLE
```

```
$ cd /var/www/html
```

13. Edit the default Nginx HTML file using a text editor such as Nano.

```
CONSOLE
```

```
$ nano index.nginx-debian.html
```

14. Change the default HTML `<h1>` tag value to a custom message such as

```
Served from Host 1.
```

```
HTML
```

```
<h1>Served by Host 1!</h1>
```

Save and close the file.

15. Restart Nginx to apply the file changes.

```
CONSOLE
```

```
$ sudo systemctl restart nginx
```

16. Disable the public network interface.

```
$ sudo ip link set enp1s0 down
```

Host 2 (Backup Web Server)

1. Access the Rocky Linux server using SSH.

CONSOLE

```
$ ssh root@10.49.144.5
```

2. Create a new non-root user.

CONSOLE

```
# adduser dbadmin
```

3. Assign the user a strong password.

CONSOLE

```
# passwd dbadmin
```

4. Add the user to the privileged `wheel` group.

CONSOLE

```
# usermod -aG wheel dbadmin
```

5. Switch to the user.

CONSOLE

```
# su - dbadmin
```

6. View the server network interfaces and verify the VPC 2.0 interface name.

CONSOLE

```
$ ip a
```

Output:

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 56:00:04:d5:77:bc brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.102/23 brd 192.0.2.255 scope global dynamic noprefixroute enp1s0
        valid_lft 85324sec preferred_lft 85324sec
    inet6 2a05:f480:3000:29a0:5400:4ff:fed5:77bc/64 scope global dynamic
noprefixroute
        valid_lft 2591636sec preferred_lft 604436sec
    inet6 fe80::5400:4ff:fed5:77bc/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: enp8s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc mq state UP group
default qlen 1000
    link/ether 5a:01:04:d5:77:bc brd ff:ff:ff:ff:ff:ff
    inet 10.49.144.5/20 brd 10.49.159.255 scope global noprefixroute enp8s0
        valid_lft forever preferred_lft forever
    inet6 fe80::5801:4ff:fed5:77bc/64 scope link
        valid_lft forever preferred_lft forever
```

7. Test access to the Vultr Load Balancer VPC 2.0 network address.

CONSOLE

```
$ ping -c 3 10.49.144.3
```

Output:

```
PING 10.49.144.3 (10.49.144.3) 56(84) bytes of data.
64 bytes from 10.49.144.3: icmp_seq=1 ttl=62 time=0.686 ms
64 bytes from 10.49.144.3: icmp_seq=2 ttl=62 time=0.731 ms
64 bytes from 10.49.144.3: icmp_seq=3 ttl=62 time=0.756 ms

--- 10.49.144.3 ping statistics ---
```

```
3 packets transmitted, 3 received, 0% packet loss, time 2054ms
rtt min/avg/max/mdev = 0.686/0.724/0.756/0.028 ms
```

8. Update the server packages.

CONSOLE

```
$ sudo dnf update
```

9. Install the Apache web server application.

CONSOLE

```
$ sudo dnf install httpd
```

10. Start the Apache web server.

CONSOLE

```
$ sudo systemctl start httpd
```

11. Configure the default Firewalld application to allow connections to the HTTP port 80 from the internal VPC 2.0 network.

CONSOLE

```
$ sudo firewall-cmd --permanent --zone=public --add-rich-
rule='
  rule family="ipv4"
  source address="10.49.144.0/20"
  port protocol="tcp" port="80" accept'
```

12. Reload Firewalld to apply the new firewall rule.

CONSOLE

```
$ sudo firewall-cmd --reload
```

13. Disable the public network interface.

```
$ sudo ip link set enp1s0 down
```

Host 3 (Backup Web Server)

1. Access the Debian server using SSH.

CONSOLE

```
$ ssh root@10.44.144.6
```

2. Create a new non-root user with sudo privileges.

CONSOLE

```
# sudo adduser webadmin && adduser webadmin sudo
```

3. Switch to the new user account.

CONSOLE

```
# su - webadmin
```

4. View the server network interfaces and verify the associated VPC 2.0 network interface.

CONSOLE

```
$ ip a
```

Output:

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
```

```
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    .....
    valid_lft forever preferred_lft forever
3: enp8s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1450 qdisc mq state UP group
default qlen 1000
    link/ether 5a:01:04:d5:77:b6 brd ff:ff:ff:ff:ff:ff
    inet 10.49.144.6/20 brd 10.49.159.255 scope global enp8s0
        valid_lft forever preferred_lft forever
    inet6 fe80::5801:4ff:fed5:77b6/64 scope link
        valid_lft forever preferred_lft forever
```

5. Test access to the Vultr Load Balancer VPC address using the Ping utility.

CONSOLE

```
$ ping 10.49.144.3
```

Output:

```
PING 10.49.144.3 (10.49.144.3) 56(84) bytes of data.
64 bytes from 10.49.144.3: icmp_seq=1 ttl=62 time=211 ms
64 bytes from 10.49.144.3: icmp_seq=2 ttl=62 time=0.868 ms
64 bytes from 10.49.144.3: icmp_seq=3 ttl=62 time=0.806 ms

--- 10.49.144.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 0.806/71.053/211.487/99.301 ms
```

6. Update the server.

CONSOLE

```
$ sudo apt update
```

7. Install Nginx.

CONSOLE

```
$ sudo apt install nginx -y
```

8. Verify the Nginx system service status.

CONSOLE

```
$ sudo systemctl status nginx
```

9. Allow connections to the HTTP port `80` from the internet VPC 2.0 network through the firewall.

CONSOLE

```
$ sudo ufw allow from 10.49.144.0/20 to any port 80
```

10. Restart the firewall to apply changes.

CONSOLE

```
$ sudo ufw reload
```

11. Navigate to the Nginx web root directory `/var/www/html`.

CONSOLE

```
$ cd /var/www/html
```

12. Edit the default Nginx file.

CONSOLE

```
$ nano index.nginx-debian.html
```

13. Change the default HTML `<h1>` tag value to `Served from Host 3`.

HTML

```
<h1>Served by Host 3!</h1>
```

Save and close the file.

14. Restart Nginx to synchronize the file changes.

CONSOLE

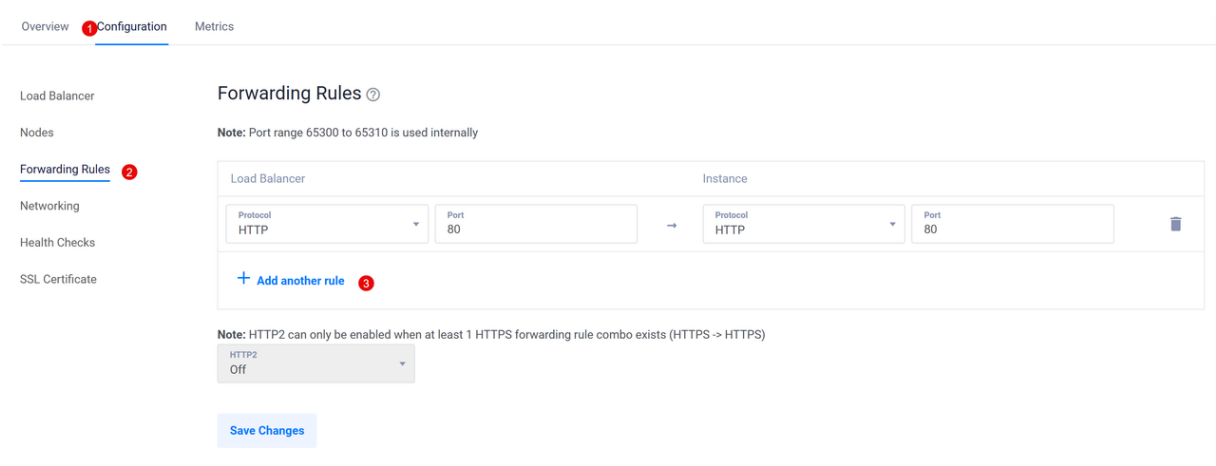
```
$ sudo systemctl restart nginx
```

15. Disable the public network interface.

```
$ sudo ip link set enp1s0 down
```

Set Up the Vultr Load Balancer Forwarding Rules to Expose VPC 2.0 Host Services

1. Access your Vultr Load Balancer instance control panel.
2. Navigate to the **Configuration** tab.
3. Click **Forwarding Rules** to access the port translation options.



The screenshot shows the Vultr Load Balancer Configuration page. The 'Configuration' tab is active, and the 'Forwarding Rules' section is selected. A note indicates that the port range 65300 to 65310 is used internally. The 'Load Balancer' section shows a rule with Protocol 'HTTP' and Port '80'. The 'Instance' section shows a rule with Protocol 'HTTP' and Port '80'. There is a '+ Add another rule' button with a red notification icon. Below the rule list, there is a note: 'Note: HTTP2 can only be enabled when at least 1 HTTPS forwarding rule combo exists (HTTPS -> HTTPS)'. The 'HTTP2' dropdown is currently set to 'Off'. A 'Save Changes' button is at the bottom.

4. Verify that the default **HTTP** port **80** rule is available, and click **Add another rule** to set up any additional port forwarding rules.
5. Click **Save Changes** to apply the Vultr Load Balancer forwarding changes.

Test Access to the VPC 2.0 Host Services

A Vultr Load Balancer accepts all external network requests and distributes traffic to the respective internal VPC network ports depending on the load-balancing strategy and forwarding rules. Follow the sections below to verify that the load balancer distributes traffic to all internal VPC 2.0 web servers.

1. Visit your domain `example.com` using a web browser such as Chrome.

```
http://example.com
```

Verify that the host 1 web page displays in your browser.

Served by Host 1!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

2. Open a new browser window and visit your domain `example.com` again.

```
http://example.com
```

Verify that the host 2 Apache default web page displays in your browser session.

Served from Host 2!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

Thank you for using nginx.

3. Open another browser window and visit the domain again.

```
http://example.com
```

Verify that the host 3 web page displays in your browser.

Serverd by Host 3!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

Thank you for using nginx.

Based on the above web server results, the load balancer distributes traffic between the main web server and the backup servers using the default `round-robin` algorithm. As a result, all web servers effectively act as one which improves the general web application availability. If one of the servers goes down, the load balancer distributes traffic to other servers within the VPC 2.0 network.

Best Practices

The Vultr Load Balancer accepts multiple connections to internal VPC 2.0 network hosts and distributes traffic based on the defined load balancing algorithm. For the best results, follow the best practices below to secure and correctly distribute traffic to all internal host servers.

- Add your domain SSL certificate to the Vultr Load Balancer to enable secure HTTPS connections to your web application. Enable the **Force HTTP to HTTPS** option within the load balancer configuration to activate automatic HTTP to HTTPS redirections.
- Set up Vultr Load Balancer firewall rules and limit access to special ports such as the SSH port to your IP address or a specific IP range.
- Configure the Vultr Load Balancer health checks to automatically perform instance checks based on your defined intervals.

Health Checks

Protocol	TCP	▼
Port	21	
Interval between health checks (in seconds)	15	▲ ▼
Response timeout (in seconds)	5	ⓘ ▲ ▼
Unhealthy Threshold	5	ⓘ ▲ ▼
Healthy Threshold	5	ⓘ ▲ ▼

[Save changes](#)

Conclusion

You have exposed VPC 2.0 host services using a Vultr Load Balancer and forwarded all external network requests to the internal network. Depending on your host services, the load balancer securely exposes all services depending on your forwarding rules. For the best results and improved server security, disable the public interfaces on each server and only enable the interface while updating or installing packages.



VULTR

