How to Configure UFW to be Port Forwarding?

written by sysadmin | 26 June 2025

The previous article explained how to configure the firewalld to become a port forwarding. This article will explain how to configure ufw applications in Ubuntu to become a port forwarding.

Problem

How to configure ufw to be port forwarding?

Solution

There are 2 methods of port forwarding: forward the connection of a port to one IP/device and forward the connection of a port to a different IP/device.

A. Forward to the same IP/device

Suppose you have an Ubuntu server with IP address 192.168.56.102 and want to close port 22 but open port 43210 if someone wants to access the server via SSH. Change the SSH port like in this article, and you have to enable ufw in the server using the command below:

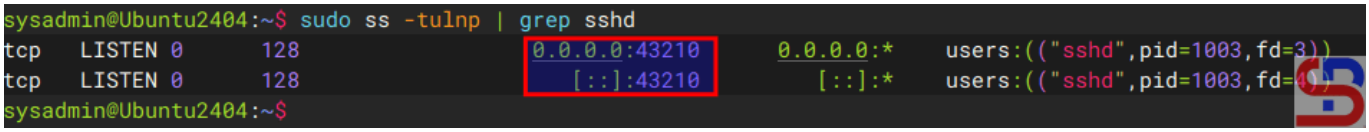
```
sudo ufw enable
```

Answer the question by pushing the **y** button. Now type the below commands to open port 22 and port 43210:

```
sudo ufw allow 43210/tcp
```

Check the SSH port using the below command and make sure the SSH port is pointed to the new port (port 43210) like in the

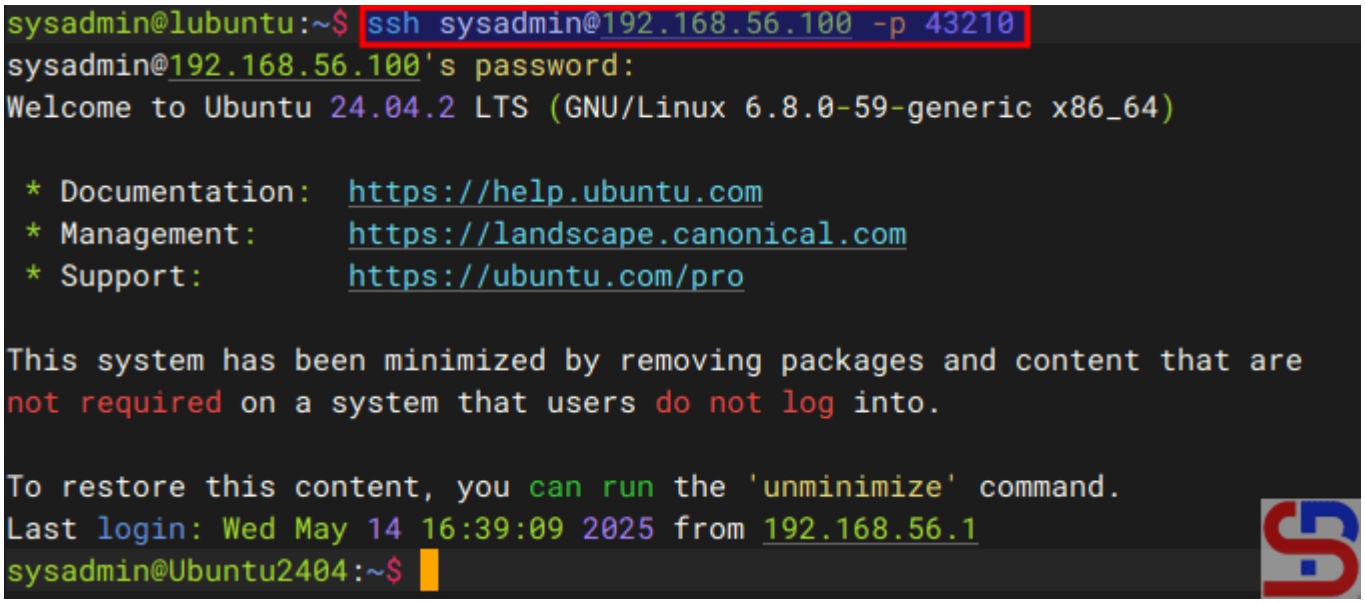
below image:



Check the port

If the port is still connected to port 22, you can go to this article to change the SSH port. Now, try to access the server using the command below:

```
ssh sysadmin@192.168.56.102 -p 43210
```



Access to the server via SSH using the port

You should access the server like in the image above. Now, you want to implement the port forwarding in the ufw so the sysadmin doesn't need to write **-p 43210** anymore. So, you have to configure the **before.rules** file in the **/etc/ufw** folder. In short, before.rules typically contains rules that handle essential network traffic before ufw's User-Defined Rules are applied. I think you have to backup the file before you configure the file using the below command:

```
sudo cp /etc/ufw/before.rules /etc/ufw/before.rules.ori
sudo vi /etc/ufw/before.rules
```

After that, copy the script below to the file **before the**

*filter section:

```
# Port forwarding from port 22 to port 43210
*nat
:PREROUTING ACCEPT [0:0]
-A PREROUTING -p tcp --dport 22 -j REDIRECT --to-port 43210
COMMIT
```

```
#
# rules.before
#
# Rules that should be run before the ufw command line added rules. Custom
# rules should be added to one of these chains:
#   ufw-before-input
#   ufw-before-output
#   ufw-before-forward
#

# Port forwarding from port 22 to port 43210
*nat
:PREROUTING ACCEPT [0:0]
-A PREROUTING -p tcp --dport 22 -j REDIRECT --to-port 43210
COMMIT

# Don't delete these required lines, otherwise there will be errors
*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
:ufw-not-local - [0:0]
# End required lines
```

Configure the before.rules file

Restart ufw using the command below:

```
sudo ufw reload
```

Now, try to access using the command below:

```
ssh sysadmin@192.168.56.102
```

You should access to the server without writing the port anymore like in the image below:

```
sysadmin@lubuntu:~$ ssh sysadmin@192.168.56.102
sysadmin@192.168.56.102's password:
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-60-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Sat May 17 08:10:38 2025 from 192.168.56.1
sysadmin@Ubuntu2404:~$
```

Access to the server without writing the port

B. Forward to the different IP/device

Suppose you have a Ubuntu server with IP address 192.168.56.102 and port 22 is available. You would like users who access the server using SSH to forward to port 22 with IP address 192.168.56.2 using RockyLinux. So, these are the steps:

1. Configure ufw

Check your Ubuntu server to see whether UFW is running on the server using the command below:

```
sudo ufw status
```

If it still doesn't run, use the command below to have ufw run on that server:

```
sudo ufw enable
```

Answer the question by pushing the **y** button. Then, open port 22 by using the command below:

```
sudo ufw allow 22/tcp
```

To run the forwarding port on UFW, you must configure the **before.rules** file in the **/etc/ufw** folder. In short, before.rules typically contains rules that handle essential network traffic before ufw's User-Defined Rules are applied. I think you have to backup the file before you configure the file using the below command:

```
sudo cp /etc/ufw/before.rules /etc/ufw/before.rules.ori
sudo vi /etc/ufw/before.rules
```

After that, copy the script below to the file **before the *filter** section:

```
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]

# Forward traffic from 192.168.56.102:22 → 192.168.56.2:22
-A PREROUTING -d 192.168.56.102 -p tcp --dport 22 -j DNAT --to-destination
192.168.56.2:22

# Masquerade outgoing traffic (adjust eth0 to your outgoing interface)
-A POSTROUTING -s 192.168.56.0/24 -o eth0 -j MASQUERADE

COMMIT
```

```
#
# rules.before
#
# Rules that should be run before the ufw command line added rules. Custom
# rules should be added to one of these chains:
#   ufw-before-input
#   ufw-before-output
#   ufw-before-forward
#

*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]

# Forward traffic from 192.168.56.102:22 → 192.168.56.2:22
-A PREROUTING -d 192.168.56.102 -p tcp --dport 22 -j DNAT --to-destination 192.168.56.2:22

# Masquerade outgoing traffic (adjust eth0 to your outgoing interface)
#-A POSTROUTING -s 192.168.56.0/24 -o enp0s8 -j MASQUERADE
-A POSTROUTING -s 192.168.56.0/24 -j MASQUERADE
COMMIT

# Don't delete these required lines, otherwise there will be errors
*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
:ufw-not-local - [0:0]
# End required lines
```

Configure the before.rules file

2. Enable IP Forwarding

Go to the **/etc/default/ufw** file and change the file from:

```
DEFAULT_FORWARD_POLICY="DROP"
```

to

```
DEFAULT_FORWARD_POLICY="ACCEPT"
```

After that, go to the **/etc/sysctl.conf** file and uncomment or add in the file:

```
net.ipv4.ip_forward=1
```

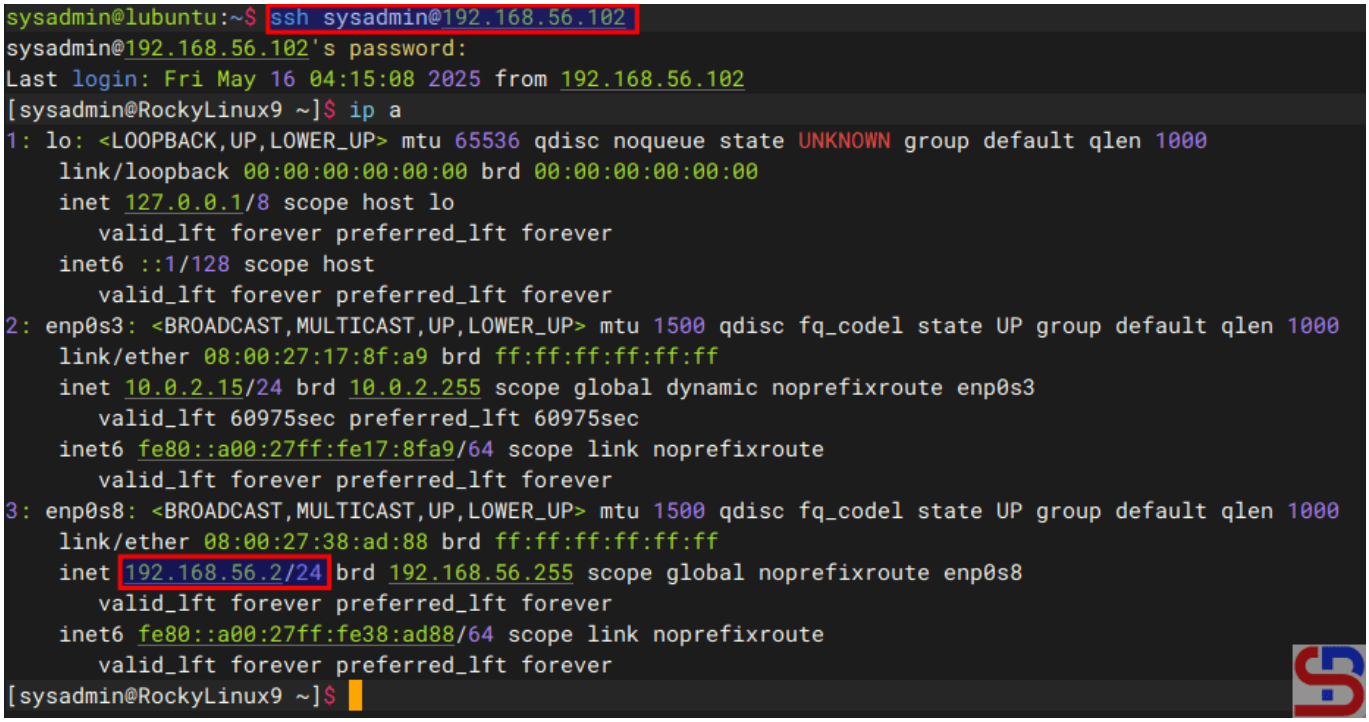
And run the below commands:

```
sudo sysctl -p
sudo ufw reload
```

3. Test the result

Now, try to access the Ubuntu server which has an IP 192.168.56.102 and you should be forwarded to the Rockylinux server that uses IP 192.168.56.2 like the below image:

```
ssh sysadmin@192.168.56.102
```



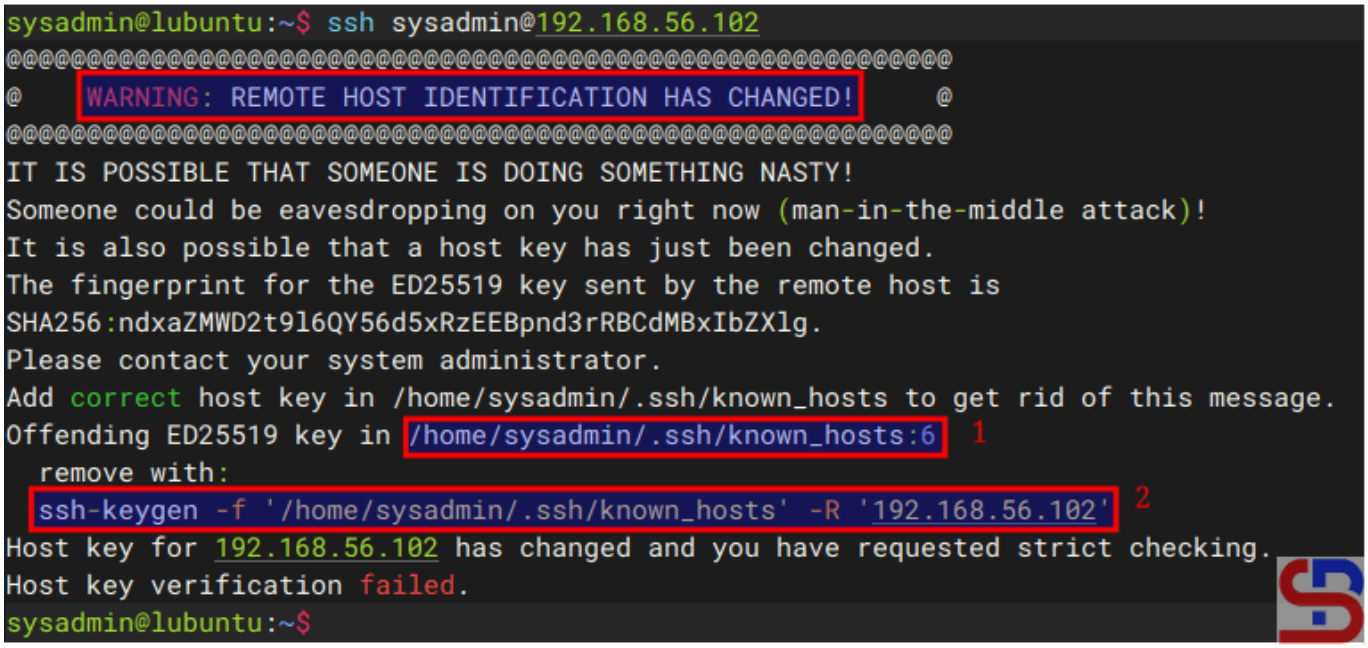
Test access

If you have a display like the image above, you have succeeded in making ufw as a forwarding port to a different IP/device.

Note

If you get an error like this:

```
WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!
```



Error when connecting the server via SSH

When you get this error, the system gives the clue to solve this error. Based on the picture above, you can go to the **/home/sysadmin/.ssh/known_hosts** file and **delete line 6** or you run the command below:

```
ssh-keygen -f '/home/sysadmin/.ssh/known_hosts' -R '192.168.56.102'
```

References

baeldung.com
gist.github.com
tecadmin.net
bobcares.com

How to Access the Server via SSH After Changing the SSH Port?

written by sysadmin | 26 June 2025

The previous article explained how to change the SSH port. Nevertheless, after I changed the port, I could not access the server via SSH using the new port.

Problem

How to access the server via SSH after changing the SSH port?

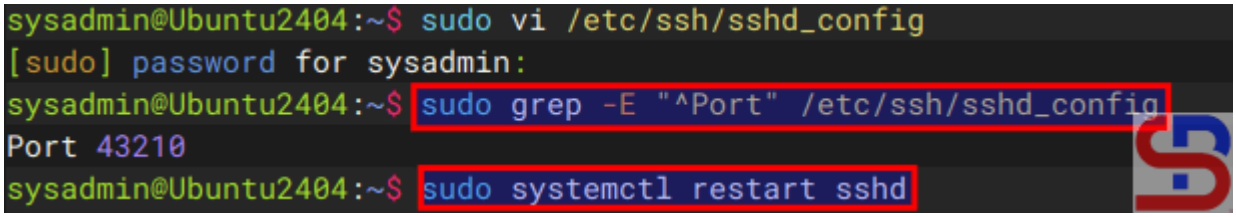
Solution

Let’s say you have changed the SSH port from **port 22 to port 43210** by changing it in the **/etc/ssh/sshd_config** file and checking the port by typing the command below:

```
sudo grep -E "^Port" /etc/ssh/sshd_config
```

After that, restart the SSH using the command below:

```
sudo systemctl restart sshd
```



Change to the new port in SSH

However, you can’t access the server via SSH using port 43210 but can still access via SSH port 22 as shown in the image below:

```
sysadmin@lubuntu:~$ ssh sysadmin@192.168.56.102 -p 43210
ssh: connect to host 192.168.56.102 port 43210: Connection refused
sysadmin@lubuntu:~$
sysadmin@lubuntu:~$ ssh sysadmin@192.168.56.102
sysadmin@192.168.56.102's password:
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-60-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Sat May 17 07:07:05 2025 from 192.168.56.1
sysadmin@Ubuntu2404:~$
```

Test the result

In the remote, type the command below to check if the SSH port has changed to Port 43210 or not:

```
sudo ss -tulnp | grep sshd
```

If you find the result as shown in the image below:

```
sysadmin@Ubuntu2404:~$ sudo grep -E "^Port" /etc/ssh/sshd_config
Port 43210
sysadmin@Ubuntu2404:~$
sysadmin@Ubuntu2404:~$ sudo ss -tulnp | grep sshd
tcp   LISTEN 0      4096                 *:22               *:*    users:(("sshd",pid=1003,fd=3),("systemd",pid=1,fd=88
sysadmin@Ubuntu2404:~$
```

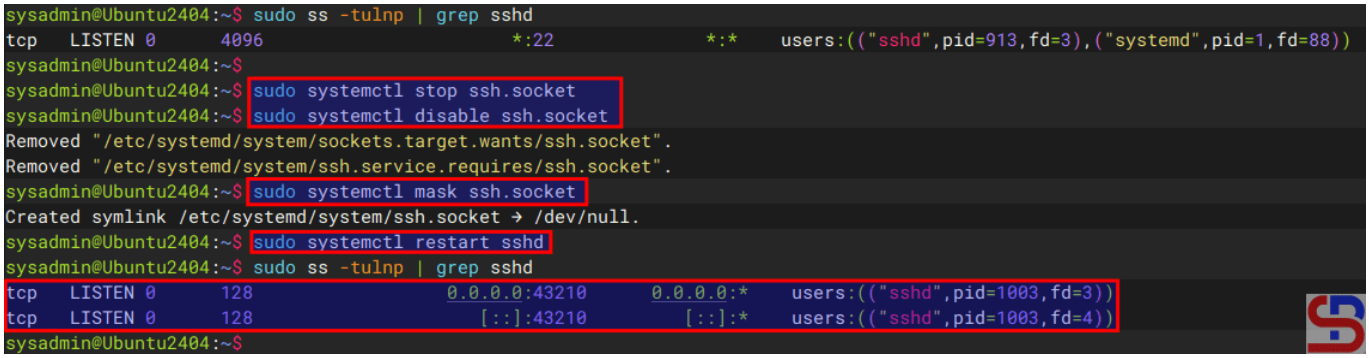
Check the port

It means the SSH is still connected to port 22 and not to port 43210. Therefore, type the commands below:

```
sudo systemctl stop ssh.socket
sudo systemctl disable ssh.socket
sudo systemctl mask ssh.socket
sudo systemctl restart sshd
```

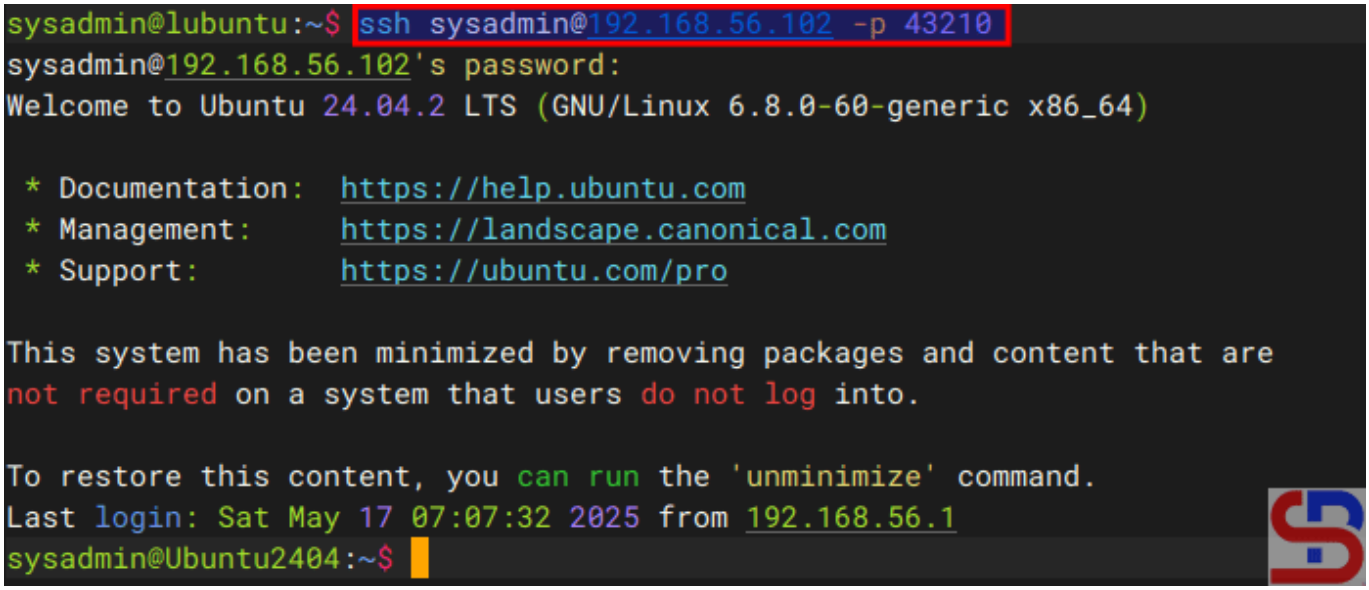
Run the previous command to check the port:

```
sudo ss -tulnp | grep sshd
```



Check the port

You can see in the image above that the SSH port has changed to port 43210 and you should be able to access the server via SSH using port 43210.



Test the result

Note

The socket statistics, or **ss**, is a tool to display network socket information. This tool has the same function as netstat but has several advantages such as faster, filtering by connection state (e.g., established, time-wait), debugging high-performance networks, and so on.

References

[askubuntu.com]

rome-rohani.medium.com
discourse.ubuntu.com
community.clearlinux.org
redhat.com

How to Configure Firewalld to be Port Forwarding?

written by sysadmin | 26 June 2025

Port forwarding is a networking technique used to redirect communication requests from one port number to another port number, typically across a network boundary such as a router or firewall. This technique can be used with Firewalld, available in RockyLinux, or derivative distros from RHEL such as AlmaLinux, CentOS, and others.

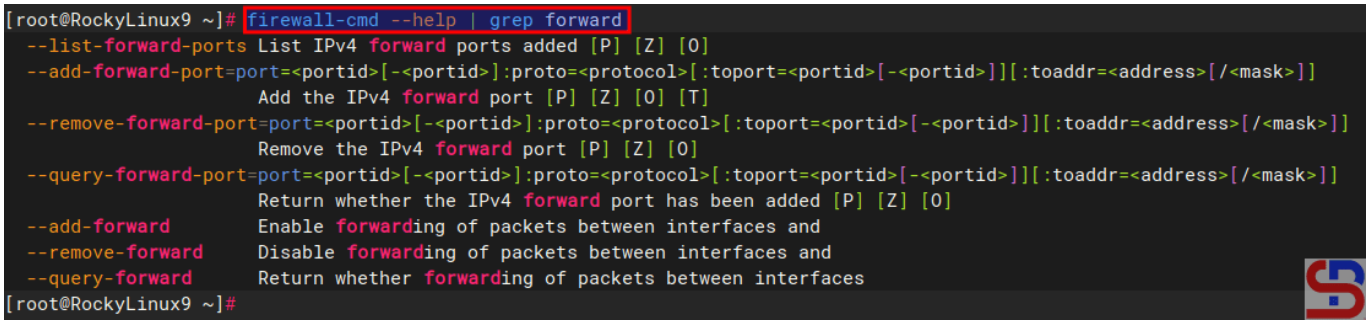
Problem

How to configure Firewalld to be port forwarding?

Solution

If you want to see the command in firewalls to run port forwarding, type the below command:

```
firewall-cmd --help | grep forward
```



The commands in firewalld for port forwarding

There are 2 methods of port forwarding: forward the connection of a port to one IP/device and forward the connection of a port to a different IP/device.

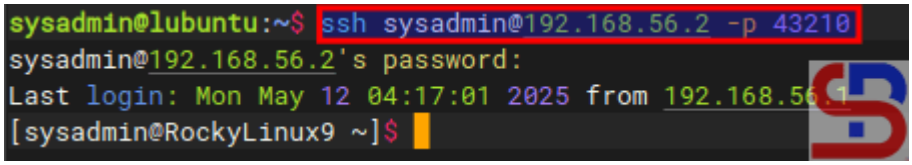
A. Forward to the same IP/device

By default, you must use the format below to forward a port in a device:

```
firewall-cmd --add-forward-port=port=port-number:proto=tcp|udp|sctp|dccp:toport=port-number
```

You can add an option **--permanent** if you want the rule to remain after reloading or rebooting the system. For example, you have a server with IP 192.168.56.2 where port 22 on the server is closed so to access the server via SSH must use port 43210. If you follow this article, then you must type the command below to access the server:

```
ssh sysadmin@192.168.56.2 -p 43210
```



Access the server via SSH using the port

However, by implementing a port forwarding you can access the server without typing the port. Let's say, the firewalld is in the device, then on the device open port 43210 using the command:

```
sudo firewall-cmd --add-port=43210/tcp --permanent
sudo firewall-cmd --reload
```

In the file **/etc/sshd/sshd_config**, change the port to be as below:

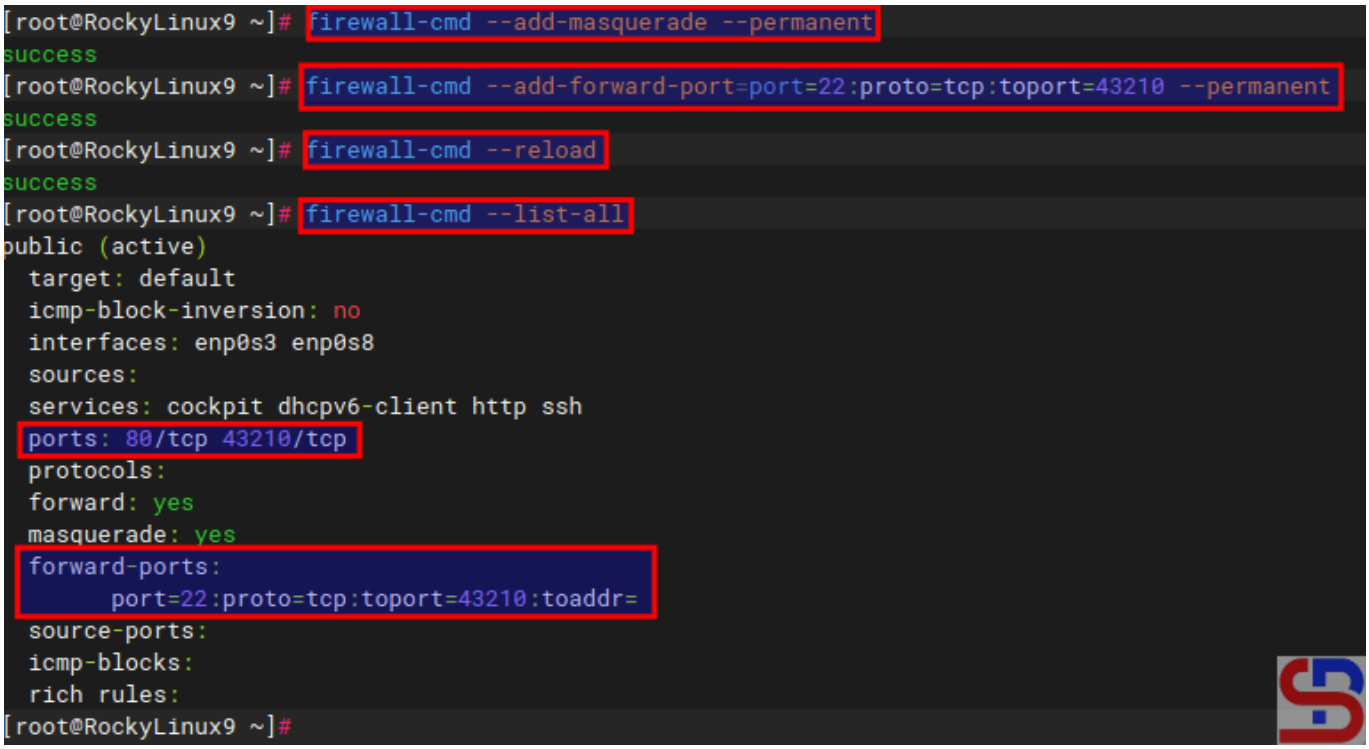
```
Port 43210
```

After that restart SSH by using the command:

```
sudo systemctl restart sshd
```

After that, type the commands below to configure the forwarding port in the firewalld:

```
firewall-cmd --add-masquerade --permanent
firewall-cmd --add-forward-port=port=22:proto=tcp:toport=43210 --permanent
firewall-cmd --reload
firewall-cmd --list-all
```

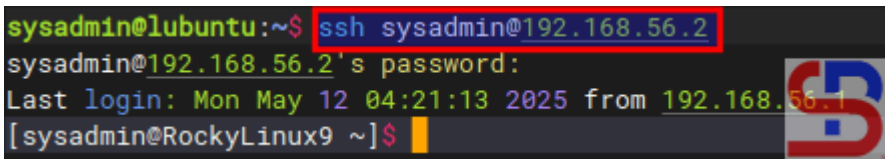


The commands to configure firewalld to be port forwarding

type the command below to access the server via SSH:

```
ssh sysadmin@192.168.56.2
```

You should be able to enter the server without having to type the 43210 port as shown below:



Access the server via SSH without writing the

port

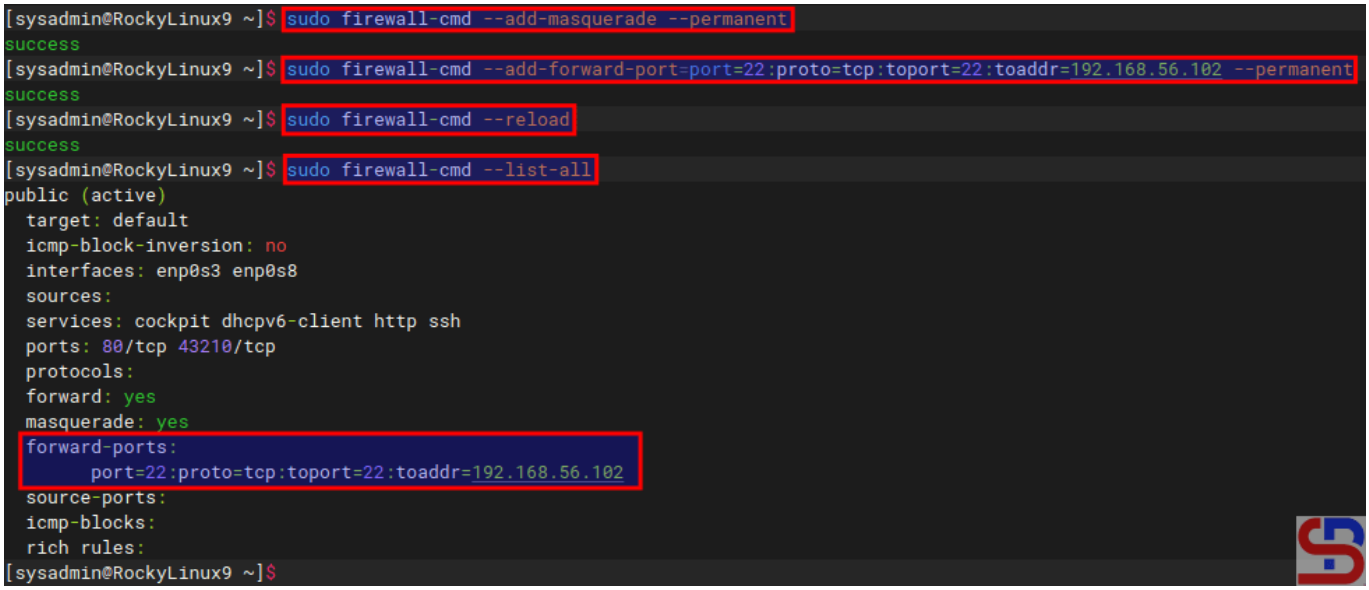
B. Forward to a different IP/device

By default, use the format below to forward a port to a different IP/device:

```
firewall-cmd --add-forward-port=port=port-number:proto=tcp|udp|sctp|dccp:toport=port-number:toaddr=ip_address
```

If you want the rule to stay in place after a system reboot or reload, you can add a **--permanent** option. As an illustration, suppose you have a server with IP address 192.168.56.2 and port 22 is available. You would like users who access port 22 to forward to port 22 with IP address 192.168.56.102. Use the command below to configure firewalls:

```
firewall-cmd --add-masquerade --permanent
sudo firewall-cmd --add-forward-port=port=22:proto=tcp:toport=22:toaddr=192.168.56.102 --permanent
firewall-cmd --reload
firewall-cmd --list-all
```

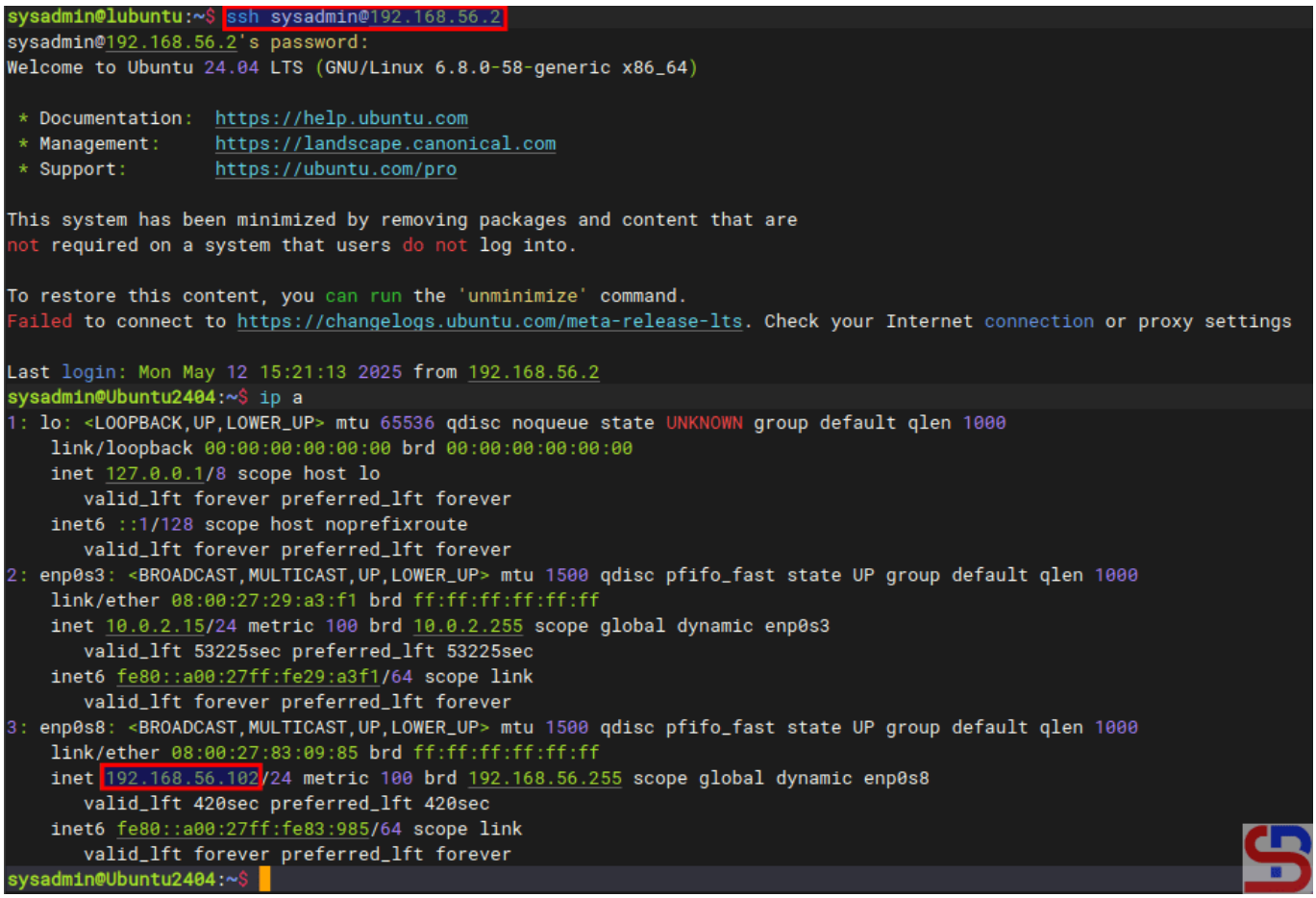


Add a forwarding port to a different IP in firewalld

If you type the command below:

```
ssh sysadmin@192.168.56.2
```

You will be forwarded to a server that uses IP 192.168.56.102 as shown below:



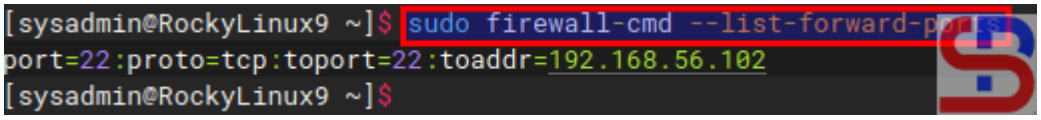
Forward a port to another IP/device

Note

To see rule forwarding is in the rule in the firewall, besides being able to use the **firewall-cmd --list-all** command, you can also use the command below:

```
sudo firewall-cmd --list-forward-ports
```

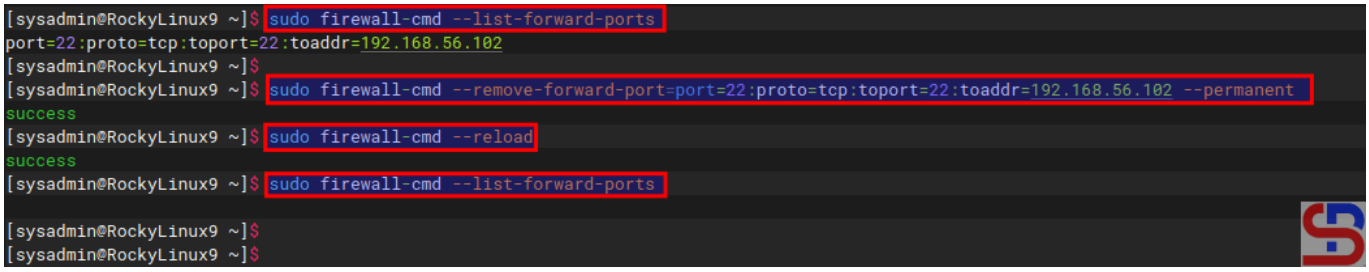
then you will see the results as shown below:



Using –list-forward-ports option

And if you want to delete a rule port forwarding in the firewall, then you can simply change the options **--add-forward-port** to **--remove-forward-port** so the command will change like in the command below:

```
sudo firewall-cmd --add-forward-
port=port=22:proto=tcp:toport=22:toaddr=192.168.56.102 --permanent
```



Remove a forwarding port rule

References

docs.redhat.com
youtube.com
musaamin.web.id
faun.pub

How to Configure Virtual Hosts in Apache on RockyLinux?

written by sysadmin | 26 June 2025

The previous article explained how to create virtual hosts in Ubuntu. This article will explain how to configure virtual hosts in Apache on Rocky Linux or derivatives of RHEL, such as Almalinux, CentOS, and so on.

Problem

How to configure virtual hosts in Apache on RockyLinux?

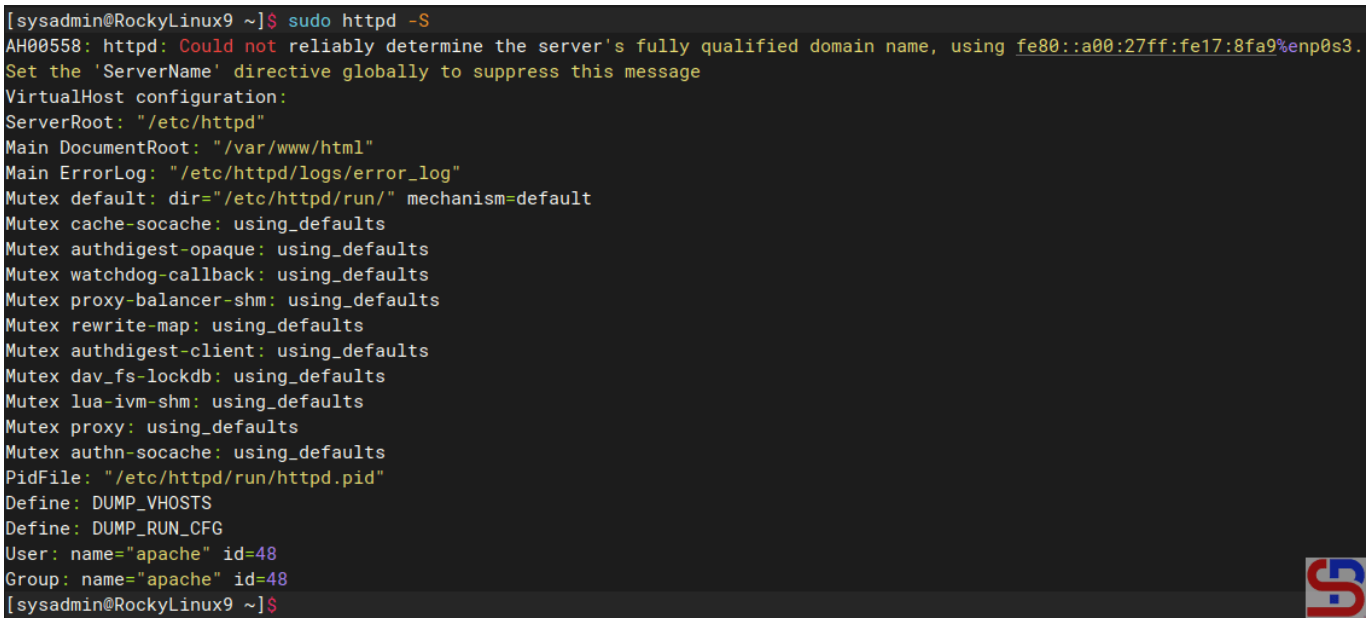
Solution

Before starting the configuration, make sure that on the RockyLinux server, the Apache application is installed by using the command:

```
yum install -y httpd
```

To see the default settings of Apache in RockyLinux, type the command below:

```
sudo httpd -S
```



Display default Apache configuration

2 types of virtual hosts can be used, name-based and IP-based, and the difference between the two can be seen in the image below:

Aspect	Name-Based	IP-Based
Definition	Uses the domain name (hostname) to distinguish between websites on the same IP.	Uses different IP addresses for each website hosted on the server.
How it Works	Server identifies the requested site by the hostname in the Host header.	Server identifies the site based on the destination IP address.
IP Requirements	Only one IP address is needed for multiple sites.	Each site requires its own unique IP address.
SSL Compatibility	May require SNI (Server Name Indication) to support multiple SSL certificates.	Easier SSL management, as each site can have its own SSL certificate.
Resource Efficiency	More efficient, as multiple sites share the same IP.	Less efficient, as each site requires its own IP.
Isolation	Sites share the same IP, so there is no isolation between them at the IP level.	Sites are isolated at the IP level, which can be beneficial for network management.
Performance	Slightly slower with SSL if SNI is not supported, otherwise similar.	No performance hit for SSL, as each site has its own IP.
Use Cases	Ideal for hosting many websites on the same server, especially for shared hosting.	Ideal for scenarios requiring multiple SSL certificates or when isolation is necessary.

Comparison of name-based and IP-based in virtual hosts

WARNING
This article uses a private IP, not a public IP.

A. name-based virtual hosts

The meaning of name-based is that you have many websites or domains, but you only have one IP. For example, you have 2 domain names: **website1.com** and **website2.com,** but you only have 1 IP, which is **192.168.56.2**. Here are the steps to get all three domains to use the same IP:

1. Create the directories and the files

By default, Apache uses the **/var/www/html** folder as its rootdocument, as shown in the image above. However, to make it easier to configure it, you should create a folder for each of these websites, as shown in the image below:

```
sudo mkdir -p /var/www/html/website1.com/
sudo mkdir -p /var/www/html/website2.com/
```

WARNING
You can change the above directory to another directory, but for the next steps, you have to follow the directory you created.

After that, create an **index.html** file for each domain:

```
sudo sh -c 'echo "<h1> This is for website1.com domain</h1>" >
/var/www/html/website1.com/index.html'
sudo sh -c 'echo "<h1> This is for website2.com domain</h1>" >
/var/www/html/website2.com/index.html'
```

2. Change ownership

Change the ownership of the folders:

```
sudo chown -R apache:apache /var/www/html/website1.com/
sudo chown -R apache:apache /var/www/html/website2.com/
sudo chmod -R 755 /var/www/html
```

3. Configuration of virtual hosts

Unlike Ubuntu and its derivatives, which use the sites-available and sites-enabled folders in configuring virtual hosts, by default, RockyLinux and its derivatives do not use both folders, but the virtual hosts configuration is inserted into the **/etc/httpd/conf.d/** folder. Therefore, type the command below to create two domains in the virtual hosts:

```
echo "<VirtualHost *:80>" | sudo tee /etc/httpd/conf.d/website1.com.conf >
/dev/null
echo "     ServerName website1.com"  | sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     ServerAlias www.website1.com"    |  sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     ServerAdmin webmaster@website1.com"   | sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     DocumentRoot /var/www/html/website1.com"  | sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     ErrorLog logs/website1-error.log"   | sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     CustomLog logs/website1-access.log combined"   | sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "</VirtualHost>"   | sudo  tee -a /etc/httpd/conf.d/website1.com.conf >
```

```
/dev/null

echo "<VirtualHost *:80>"  | sudo tee /etc/httpd/conf.d/website2.com.conf >
/dev/null
echo "     ServerName website2.com"  | sudo  tee -a
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "     ServerAlias www.website2.com"  | sudo  tee -a
/etc/httpd/conf.d/website2.com.com >  /dev/null
echo "     ServerAdmin webmaster@website2.com"  | sudo  tee -a
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "     DocumentRoot /var/www/html/website2.com"   | sudo  tee -a
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "     ErrorLog logs/website2-error.log"   | sudo  tee -a
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "     CustomLog logs/website2-access.log combined"   | sudo  tee -a
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "</VirtualHost>"   | sudo  tee -a /etc/httpd/conf.d/website2.com.conf >
/dev/null
```

WARNING

You can change ***:80** to your IP server like **192.168.56.2:80**.

4. Check the configuration

Use the command below to check whether there is an Apache configuration that is an error or not by using the command below:

```
sudo apachectl configtest
```

If there is no error, then reload Apache using the command below:

```
sudo systemctl reload httpd
```

WARNING

Use the command above if there is a change in the configuration of virtual hosts in each domain.

5. Check in the browser

Because this article uses a private IP, you must configure it in the hosts file before you check the browser. If you use Windows, change the hosts file in **C:\Windows\System32\drivers\etc\hosts** or in **/etc/hosts** if you use Linux. In the hosts file, add the below script:

```
192.168.56.2  website1.com website2.com
```

Info
Change IP 192.168.56.2 with your RockyLinux IP server.

By default, Rockylinux activates the firewall, so you have to open the HTTP port using the command below:

```
firewall-cmd --add-service=http --permanent
firewall-cmd --reload
```

Open your browser and type each of these domains, then there should be a site displayed as in the image below:

http://website1.com

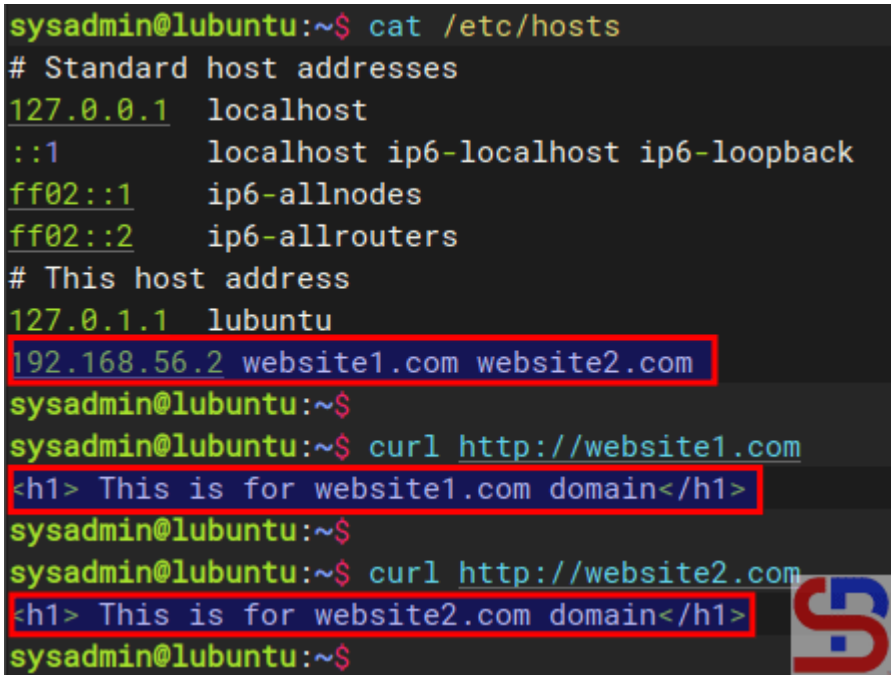
Site website1.com

http://website2.com

site website2.com

If you use Linux, you can use the command below to check the result:

```
curl http://website1.com
curl http://website2.com
```



Using the curl command

By default, websites work on the web server using port 80. But you can change port 80 to another port as long as the port is not used on the server. For example, if you want the website1.com site to use port **8080**, change the **/etc/httpd/conf.d/website1.com.conf** file and change its contents to something like this:

```
Listen 8080
<VirtualHost *:8080>
     ServerName website1.com
     ServerAlias www.website1.com
```

```
    ServerAdmin webmaster@website1.com
    DocumentRoot /var/www/html/website1.com
    ErrorLog logs/website1-error.log
    CustomLog logs/website1-access.log combined
</VirtualHost>
```

Don’t forget to open the 8080 port on the Rockylinux server and reload Apache using the command below:

```
firewall-cmd --add-port=8080/tcp --permanent
firewall-cmd --reload
sudo systemctl reload apache
```

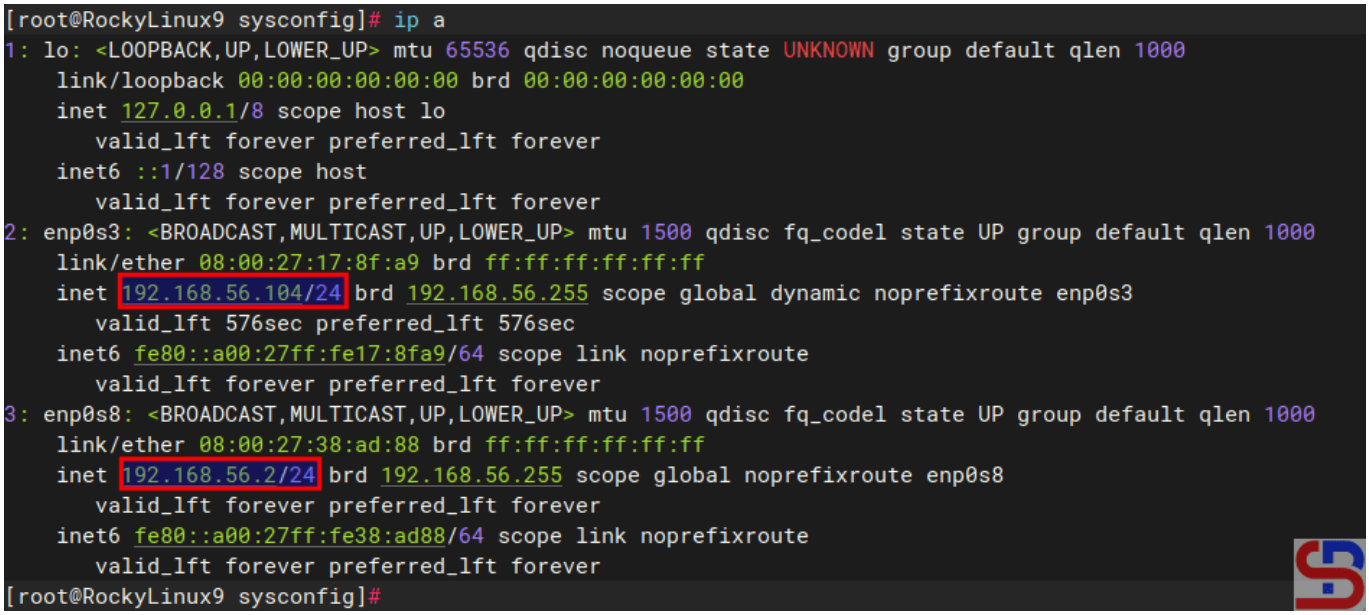
Open your browser and type the command below:

```
http://website1.com:8080
```


```
Site website1.com:8080
```

B. IP-based virtual hosts

The meaning of IP-based is that you use a different IP address for each website. For example, you have 2 IPs and 2 domains, where IP **192.168.56.2 is for site1.com,** and IP **192.168.56.104 is for site2.com**. This article will use a server that has 2 IPs, as shown below:



Using 2 NICs in a server

1. Create the directories and the files

By default, Apache uses the **/var/www/html** folder as its rootdocument, as shown in the image above. However, to make it easier to configure it, you should create a folder for each of these websites, as shown in the image below:

```
sudo mkdir -p /var/www/html/site1.com/
sudo mkdir -p /var/www/html/site2.com/
```

WARNING

You can change the above directory to another directory, but for the next steps, you have to follow the directory you created.

After that, create an index.html file for each domain:

```
sudo sh -c 'echo "<h1> This is for site1.com domain</h1>" >
/var/www/html/site1.com/index.html'
sudo sh -c 'echo "h1> This is for site2.com domain</h1>" >
/var/www/html/site2.com/index.html'
```

2. Change ownership

Change the ownership of the folders:

```
sudo chown -R apache:apache /var/www/html/site1.com/
sudo chown -R apache:apache /var/www/html/site2.com/
sudo chmod -R 755 /var/www/html
```

3. Configuration of virtual hosts

Unlike Ubuntu and its derivatives, which use the sites-available and sites-enabled folders in configuring virtual hosts, by default, RockyLinux and its derivatives do not use both folders, but the virtual hosts configuration is inserted into the **/etc/httpd/conf.d/** folder. Therefore, type the command below to create two domains in the virtual hosts:

```
echo "<VirtualHost 192.168.56.2:80>" | sudo tee
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     ServerName website1.com"  | sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     ServerAlias www.website1.com"    |  sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     ServerAdmin webmaster@website1.com"   | sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     DocumentRoot /var/www/html/website1.com"  | sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     ErrorLog logs/website1-error.log"   | sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "     CustomLog logs/website1-access.log combined"   | sudo  tee -a
/etc/httpd/conf.d/website1.com.conf > /dev/null
echo "</VirtualHost>"   | sudo  tee -a /etc/httpd/conf.d/website1.com.conf >
/dev/null

echo "<VirtualHost 192.168.56.104:80>"  | sudo tee
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "     ServerName website2.com"  | sudo  tee -a
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "     ServerAlias www.website2.com"  | sudo  tee -a
/etc/httpd/conf.d/website2.com.com >  /dev/null
echo "     ServerAdmin webmaster@website2.com"  | sudo  tee -a
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "     DocumentRoot /var/www/html/website2.com"   | sudo  tee -a
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "     ErrorLog logs/website2-error.log"   | sudo  tee -a
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "     CustomLog logs/website2-access.log combined"   | sudo  tee -a
/etc/httpd/conf.d/website2.com.conf > /dev/null
echo "</VirtualHost>"   | sudo  tee -a /etc/httpd/conf.d/website2.com.conf >
```

```
/dev/null
```

4. Check the configuration

Use the command below to check whether there is an Apache configuration that is an error or not by using the command below:

```
sudo apachectl configtest
```

If there is no error, then reload Apache using the command below:

```
sudo systemctl reload httpd
```

WARNING
Use the command above if there is a change in the configuration of virtual hosts in each domain.

5. Check in the browser

Because this article uses a private IP, you must configure it in the hosts file before you check the browser. If you use Windows, change the hosts file in **C:\Windows\System32\drivers\etc\hosts** or in **/etc/hosts** if you use Linux. In the hosts file, add the below script:

```
192.168.56.2   site1.com
192.168.56.104 site2.com
```

Info
Change IP 192.168.56.2 & IP 192.168.56.104 with your RockyLinux IP server.

By default, Rockylinux activates the firewall, so you have to open the HTTP port using the command below:

```
firewall-cmd --add-service=http --permanent
```

```
firewall-cmd --reload
```

Open your browser and type each of these domains then there should be a site displayed as in the image below:

```
http://site1.com
```


```
Site site1.com
```

```
http://site2.com
```


```
Site site2.com
```

If you use Linux, you can use the command below to check the result:

```
curl http://site1.com
curl http://site2.com
```

```
sysadmin@lubuntu:~$ cat /etc/hosts
# Standard host addresses
127.0.0.1  localhost
::1        localhost ip6-localhost ip6-loopback
ff02::1    ip6-allnodes
ff02::2    ip6-allrouters
# This host address
127.0.1.1  lubuntu
192.168.56.2 site1.com
192.168.56.104 site2.com
sysadmin@lubuntu:~$
sysadmin@lubuntu:~$ curl http://site1.com
<h1> This is for website1.com domain</h1>
sysadmin@lubuntu:~$
sysadmin@lubuntu:~$ curl http://site2.com
<h1> This is for site2.com domain</h1>
sysadmin@lubuntu:~$
```

Using the curl command

By default, websites work on the web server using port 80. But you can change port 80 to another port as long as the port is not used on the server. So, if you want the site1.com site to use port **8181**, change the **/etc/httpd/conf.d/website1.com.conf** file and change its contents to something like this:

```
Listen 8181
<VirtualHost 192.168.56.102:8181>
     ServerName site1.com
     ServerAlias www.site1.com
     ServerAdmin webmaster@site1.com
     DocumentRoot /var/www/html/site1.com
     ErrorLog logs/site1-error.log CustomLog logs/site1-access.log combined
</VirtualHost>
```

If you use the firewall in your Ubuntu server, don’t forget to open port 8181 using the command below:

```
sudo firewall-cmd --add-port=8181/tcp --permanent
sudo firewall-cmd --reload
sudo systemctl reload apache
```

Reload Apache and open it in the browser by typing the

command:

```
http://site1.com:8181
```


```
Site site1.com:8181
```

Note

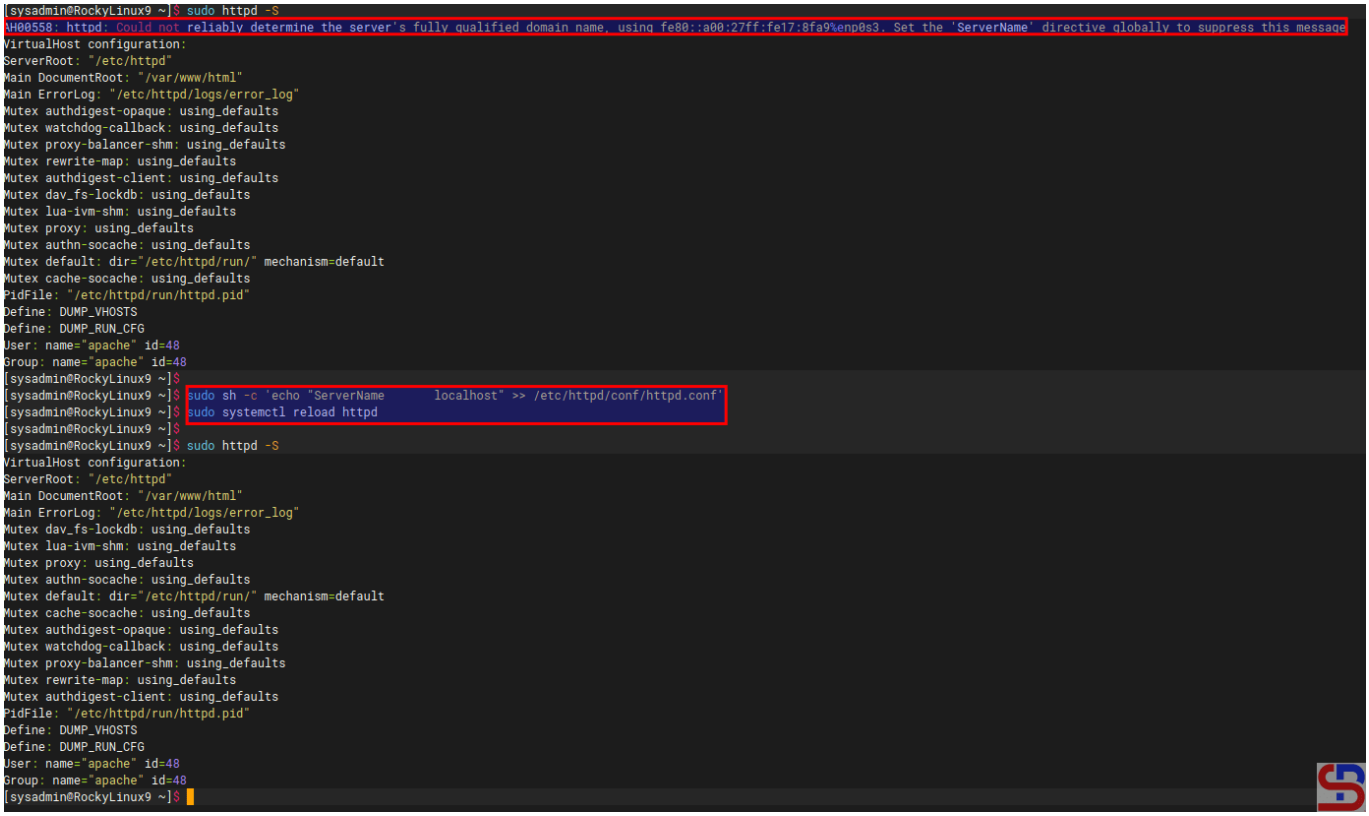
If you want to remove the error like this:

```
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 192.168.56.103. Set the 'ServerName' directive globally to suppress this message
```

Go to the **/etc/httpd/conf/httpd.conf** and insert the script below:

```
ServerName localhost
```

Reload the Apache, and the error will disappear, like in the image below:



Remove error AH00558

References

sandeepbansod.medium.com
reintech.io
tecmint.com
httpd.apache.org
askubuntu.com
dewaweb.com
arubacloud.com

How to Configure Virtual Hosts in Apache on Ubuntu?

written by sysadmin | 26 June 2025

Virtual hosts are a feature on a web server, such as Apache or Nginx, to run more than one site on a server. By using this feature, you can easily configure multiple domains on a

server and save on operational costs because you only need one server or a public IP. This article will explain how to configure virtual hosts in Apache on Ubuntu.

Problem

How to configure virtual hosts in Apache on Ubuntu?

Solution

Before starting the configuration, make sure that on the Ubuntu server, the Apache application is installed by using the command:

```
apt update
apt install -y apache2
```

To see the default settings of Apache in Ubuntu, type the command below:

```
sudo apache2ctl -S
```

Display default Apache configuration

2 types of virtual hosts can be used, name-based and IP-based, and the difference between the two can be seen in the image below:

Aspect	Name-Based	IP-Based
Definition	Uses the domain name (hostname) to distinguish between websites on the same IP.	Uses different IP addresses for each website hosted on the server.
How it Works	Server identifies the requested site by the hostname in the Host header.	Server identifies the site based on the destination IP address.
IP Requirements	Only one IP address is needed for multiple sites.	Each site requires its own unique IP address.
SSL Compatibility	May require SNI (Server Name Indication) to support multiple SSL certificates.	Easier SSL management, as each site can have its own SSL certificate.
Resource Efficiency	More efficient, as multiple sites share the same IP.	Less efficient, as each site requires its own IP.
Isolation	Sites share the same IP, so there is no isolation between them at the IP level.	Sites are isolated at the IP level, which can be beneficial for network management.
Performance	Slightly slower with SSL if SNI is not supported, otherwise similar.	No performance hit for SSL, as each site has its own IP.
Use Cases	Ideal for hosting many websites on the same server, especially for shared hosting.	Ideal for scenarios requiring multiple SSL certificates or when isolation is necessary.

Comparison of name-based and IP-based in virtual hosts

WARNING

This article uses a private IP, not a public IP.

A. name-based virtual hosts

The meaning of name-based is that you have many websites or domains, but you only have one IP. For example, you have 2 domain names: **website1.com** and **website2.com,** but you only have 1 IP, which is **192.168.56.100**. Here are the steps to get all three domains to use the same IP:

1. Create the directories and the files

By default, Apache uses the **/var/www/html** folder as its rootdocument, as shown in the image above. However, to make it easier to configure it, you should create a folder for each of these websites, as shown in the image below:

```
sudo mkdir -p /var/www/html/website1.com/
sudo mkdir -p /var/www/html/website2.com/
```

WARNING
You can change the above directory to another directory, but for the next steps, you have to follow the directory you created.

After that, create an index.html file for each domain:

```
sudo sh -c 'echo "<h1> This is for website1.com domain</h1>" >
/var/www/html/website1.com/index.html'
sudo sh -c 'echo "<h1> This is for website2.com domain</h1>" >
/var/www/html/website2.com/index.html'
```

2. Change ownership

Change the ownership of the folders:

```
sudo chown -R www-data:www-data /var/www/html/website1.com/
sudo chown -R www-data:www-data /var/www/html/website2.com/
sudo chmod -R 755 /var/www/html
```

3. Configuration of virtual hosts

By default, 2 directories are used to manage the many domains in the virtual hosts running on that server: the **sites-available** and **sites-enabled** directories located in the **/etc/apache2** directory. The sites-enabled directory contains all the configurations of the website (virtual host) that are available on the server but are not yet activated automatically. In contrast, the sites-enabled directory contains a symlink (symbolic link) to the configuration file that exists in the sites-available directory, and only the files that exist in the sites-enabled directory will be executed and activated by the web server if the web server is restarted or reloaded. Use the command below to create two websites on virtual hosts:

```
echo '<VirtualHost *:80>' | sudo tee /etc/apache2/sites-
available/website1.com.conf > /dev/null
echo '    ServerName website1.com' | sudo tee -a /etc/apache2/sites-
available/website1.com.conf > /dev/null
echo '    ServerAlias www.website1.com' | sudo tee -a /etc/apache2/sites-
available/website1.com.conf > /dev/null
echo '    ServerAdmin webmaster@website1.com' | sudo tee -a
/etc/apache2/sites-available/website1.com.conf > /dev/null
```

```
echo '    DocumentRoot /var/www/html/website1.com' | sudo tee -a
/etc/apache2/sites-available/website1.com.conf > /dev/null
echo '    ErrorLog ${APACHE_LOG_DIR}/website1-error.log' | sudo tee -a
/etc/apache2/sites-available/website1.com.conf > /dev/null
echo '    CustomLog ${APACHE_LOG_DIR}/website1-access.log combined' | sudo
tee -a /etc/apache2/sites-available/website1.com.conf > /dev/null
echo '</VirtualHost>' | sudo tee -a /etc/apache2/sites-
available/website1.com.conf > /dev/null

echo '<VirtualHost *:80>' | sudo tee /etc/apache2/sites-
available/website2.com.conf > /dev/null
echo '    ServerName website2.com' | sudo tee -a /etc/apache2/sites-
available/website2.com.conf > /dev/null
echo '    ServerAlias www.website2.com' | sudo tee -a /etc/apache2/sites-
available/website2.com.conf > /dev/null
echo '    ServerAdmin webmaster@website2.com' | sudo tee -a
/etc/apache2/sites-available/website2.com.conf > /dev/null
echo '    DocumentRoot /var/www/html/website2.com' | sudo tee -a
/etc/apache2/sites-available/website2.com.conf > /dev/null
echo '    ErrorLog ${APACHE_LOG_DIR}/website2-error.log' | sudo tee -a
/etc/apache2/sites-available/website2.com.conf > /dev/null
echo '    CustomLog ${APACHE_LOG_DIR}/website2-access.log combined' | sudo
tee -a /etc/apache2/sites-available/website2.com.conf > /dev/null
echo '</VirtualHost>' | sudo tee -a /etc/apache2/sites-
available/website2.com.conf > /dev/null
```

WARNING
You can change ***:80** to your IP server like **192.168.56.102:80**.

Then type the command below to enable the Virtual Hosts configuration:

```
sudo a2ensite website1.com.conf
sudo a2ensite website2.com.conf
```

Type the command below to disable the default virtual hosts configuration:

```
sudo a2dissite 000-default.conf
```

WARNING
If you want to change the configuration of virtual hosts, you have to **change**

it in the sites-available directory and not in the sites-enabled directory.

5. Check the configuration

Use the command below to check whether there is an Apache configuration that is an error or not by using the command below:

```
sudo apache2ctl configtest
```

If there is no error, then reload Apache using the command below:

```
sudo systemctl reload apache2
```

WARNING
Use the command above if there is a change in the configuration of virtual hosts in each domain.

6. Check in the browser

Because this article uses a private IP, you must configure it in the hosts file before you check the browser. If you use Windows, change the hosts file in **C:\Windows\System32\drivers\etc\hosts** or in **/etc/hosts** if you use Linux. In the hosts file, add the below script:

```
192.168.56.102  website1.com website2.com
```

Info
Change IP 192.168.56.102 with your Ubuntu IP server.

If your Ubuntu server uses a firewall, type the command below to open the port for Apache:

```
sudo ufw allow 'Apache Full'
```

Open your browser and type each of these domains, then there should be a site displayed as in the image below:

http://website1.com

Site website1.com

http://website2.com

site website2.com

If you use Linux, you can use the command below to check the result:

```
curl http://website1.com
curl http://website2.com
```

```
sysadmin@lubuntu:~$ cat /etc/hosts
# Standard host addresses
127.0.0.1  localhost
::1        localhost ip6-localhost ip6-loopback
ff02::1    ip6-allnodes
ff02::2    ip6-allrouters
# This host address
127.0.1.1  lubuntu
192.168.56.102 site1.com
192.168.56.103 site2.com
sysadmin@lubuntu:~$
sysadmin@lubuntu:~$ curl http://site1.com
<h1> This is for site1.com domain</h1>
sysadmin@lubuntu:~$
sysadmin@lubuntu:~$ curl http://site2.com
<h1> This is for site2.com domain</h1>
sysadmin@lubuntu:~$
```

Using the curl command

By default, websites work on the web server using port 80. But you can change port 80 to another port as long as the port is not used on the server. For example, if you want the website1.com site to use port **8080**, change the **/etc/apache2/sites-available/website1.com.conf** file and change its contents to something like this:

```
Listen 8080
<VirtualHost *:8080>
     ServerName website1.com
     ServerAlias www.website1.com
     ServerAdmin webmaster@website1.com
     DocumentRoot /var/www/html/website1.com
     ErrorLog ${APACHE_LOG_DIR}/website1-error.log
     CustomLog ${APACHE_LOG_DIR}/website1-access.log combined
</VirtualHost>
```

If you use the firewall in the Ubuntu server, don’t forget to open port 8080 using the command below:

```
sudo ufw allow 8080
```

Reload Apache and open it in the browser by typing the command:

http://website1.com:8080

Site website1.com:8080

B. IP-based virtual hosts

The meaning of IP-based is that you use a different IP address for each website. For example, you have 2 IPs and 2 domains, where IP **192.168.56.102 is for site1.com,** and IP **192.168.56.103 is for site2.com**. This article will use a server that has 2 IPs, as shown below:

```
sysadmin@Ubuntu2404:~$ ip a
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 noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:29:a3:f1 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.103/24 metric 100 brd 192.168.56.255 scope global dynamic enp0s3
       valid_lft 573sec preferred_lft 573sec
    inet6 fe80::a00:27ff:fe29:a3f1/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:83:09:85 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.102/24 metric 100 brd 192.168.56.255 scope global dynamic enp0s8
       valid_lft 565sec preferred_lft 565sec
    inet6 fe80::a00:27ff:fe83:985/64 scope link
       valid_lft forever preferred_lft forever
sysadmin@Ubuntu2404:~$
```

Using 2 NICs in a server

1. Create the directories and the files

By default, Apache uses the **/var/www/html** folder as its rootdocument, as shown in the image above. However, to make it easier to configure it, you should create a folder for each of these websites, as shown in the image below:

```
sudo mkdir -p /var/www/html/site1.com/
sudo mkdir -p /var/www/html/site2.com/
```

WARNING
You can change the above directory to another directory, but for the next steps, you have to follow the directory you created.

After that, create an index.html file for each domain:

```
sudo sh -c 'echo "<h1> This is for site1.com domain</h1>" >
/var/www/html/site1.com/index.html'
sudo sh -c 'echo "<h1> This is for site2.com domain</h1>" >
/var/www/html/site2.com/index.html'
```

2. Change ownership

Change the ownership of the folders:

```
sudo chown -R www-data:www-data /var/www/html/site1.com/
sudo chown -R www-data:www-data /var/www/html/site2.com/
sudo chmod -R 755 /var/www/html
```

3. Configuration of virtual hosts

By default, 2 directories are used to manage the many domains in the virtual hosts running on that server: the **sites-available** and **sites-enabled** directories located in the **/etc/apache2** directory. The sites-enabled directory contains all the configuration of the website (virtual host) that is available on the server, but is not yet activated automatically while the sites-enabled directory contains a symlink (symbolic link) to the configuration file that exists in the sites-available directory and only the files that exist in the site-enabled directory will be executed and activated by the web server if the webserver is restarted or reloaded. Use the command below to create a virtual hosts directory:

```
echo '<VirtualHost 192.168.56.102:80>' | sudo tee /etc/apache2/sites-
available/site1.com.conf > /dev/null
echo '    ServerName site1.com' | sudo tee -a /etc/apache2/sites-
available/site1.com.conf > /dev/null
echo '    ServerAlias www.site1.com' | sudo tee -a /etc/apache2/sites-
```

```
available/site1.com.conf > /dev/null
echo '    ServerAdmin webmaster@site1.com' | sudo tee -a /etc/apache2/sites-
available/site1.com.conf > /dev/null
echo '    DocumentRoot /var/www/html/site1.com' | sudo tee -a
/etc/apache2/sites-available/site1.com.conf > /dev/null
echo '    ErrorLog ${APACHE_LOG_DIR}/site1-error.log' | sudo tee -a
/etc/apache2/sites-available/site1.com.conf > /dev/null
echo '    CustomLog ${APACHE_LOG_DIR}/site1-access.log combined' | sudo tee -
a /etc/apache2/sites-available/site1.com.conf > /dev/null
echo '</VirtualHost>' | sudo tee -a /etc/apache2/sites-
available/site1.com.conf > /dev/null

echo '<VirtualHost 192.168.56.103:80>' | sudo tee /etc/apache2/sites-
available/site2.com.conf > /dev/null
echo '    ServerName site2.com' | sudo tee -a /etc/apache2/sites-
available/site2.com.conf > /dev/null
echo '    ServerAlias www.site2.com' | sudo tee -a /etc/apache2/sites-
available/site2.com.conf > /dev/null
echo '    ServerAdmin webmaster@site2.com' | sudo tee -a /etc/apache2/sites-
available/site2.com.conf > /dev/null
echo '    DocumentRoot /var/www/html/site2.com' | sudo tee -a
/etc/apache2/sites-available/site2.com.conf > /dev/null
echo '    ErrorLog ${APACHE_LOG_DIR}/site2-error.log' | sudo tee -a
/etc/apache2/sites-available/site2.com.conf > /dev/null
echo '    CustomLog ${APACHE_LOG_DIR}/site2-access.log combined' | sudo tee -
a /etc/apache2/sites-available/site2.com.conf > /dev/null
echo '</VirtualHost>' | sudo tee -a /etc/apache2/sites-
available/site2.com.conf > /dev/null
```

WARNING

If you want to change the configuration of virtual hosts, you have to **change it in the sites-available directory** and not in the sites-enabled directory.

Then type the command below to enable the Virtual Hosts configuration:

```
sudo a2ensite site1.com.conf
sudo a2ensite site2.com.conf
```

Type the command below to disable the default virtual hosts configuration:

```
sudo a2dissite 000-default.conf
```

5. Check the configuration

Use the command below to check whether there is an Apache configuration that is an error or not by using the command below:

```
sudo apache2ctl configtest
```

If there is no error, then reload Apache using the command below:

```
sudo systemctl reload apache2
```

WARNING
Use the command above if there is a change in the configuration of virtual hosts in each domain.

6. Check in the browser

Because this article uses a private IP, you must configure it in the hosts file before you check the browser. If you use Windows, change the hosts file in **C:\Windows\System32\drivers\etc\hosts** or in **/etc/hosts** if you use Linux. In the hosts file, add the below script:

```
192.168.56.102  site1.com
192.168.56.103   site2.com
```

Info
Change IP 192.168.56.102 and IP 192.168.56.103 with your Ubuntu IP server.

If your Ubuntu server uses a firewall, type the command below to open the port for Apache:

```
sudo ufw allow 'Apache Full'
```

Open your browser and type each of these domains, then there should be a site displayed as in the image below:

http://site1.com

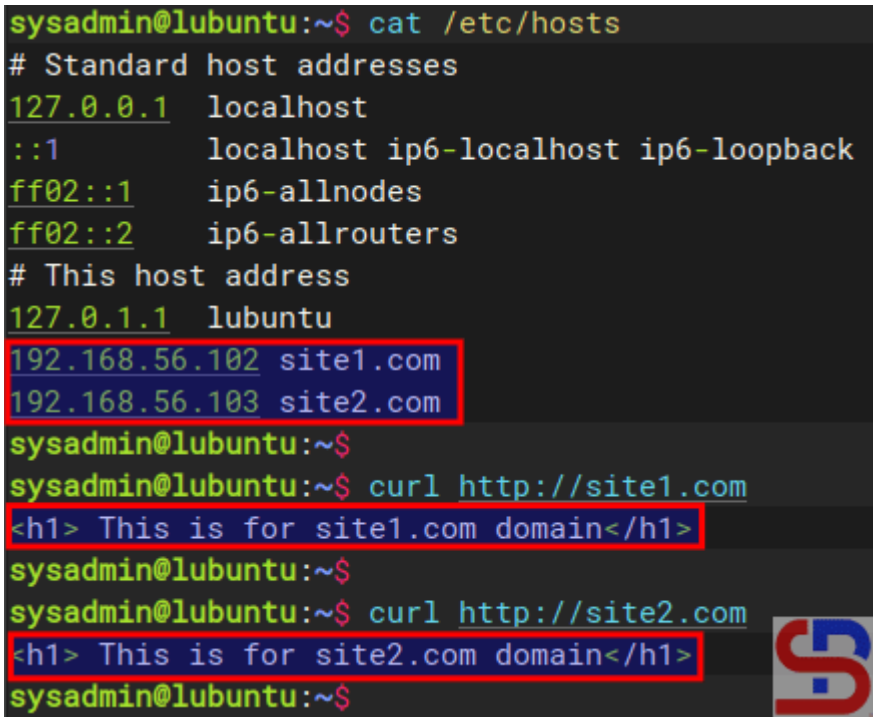
Site site1.com

http://site2.com

Site site2.com

If you use Linux, you can use the command below to check the result:

```
curl http://site1.com
curl http://site2.com
```



Using the curl command

By default, websites work on the web server using port 80. But you can change port 80 to another port as long as the port is not used on the server. So, if you want the site1.com site to use port **8181**, change the **/etc/apache2/sites-available/site1.com.conf** file and change its contents to something like this:

```
Listen 8181
<VirtualHost 192.168.56.102:8181>
     ServerName site1.com
     ServerAlias www.site1.com
     ServerAdmin webmaster@site1.com
     DocumentRoot /var/www/html/site1.com
     ErrorLog ${APACHE_LOG_DIR}/site1-error.log
     CustomLog ${APACHE_LOG_DIR}/site1-access.log combined
</VirtualHost>
```

If you use the firewall in your Ubuntu server, don’t forget to open port 8181 using the command below:

```
sudo ufw allow 8181
```

Reload Apache and open it in the browser by typing the command:

```
http://site1.com:8181
```


```
Site site1.com:8181
```

Note

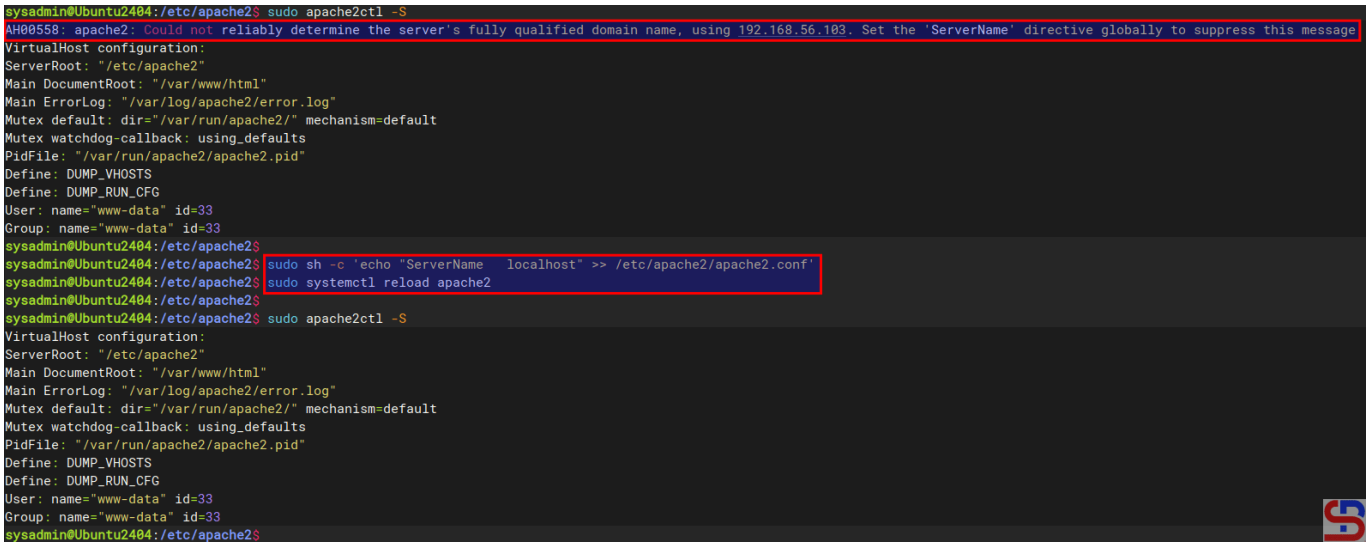
If you want to remove the error like this:

```
AH00558: apache2: Could not reliably determine the server’s fully qualified domain name, using 192.168.56.103. Set the ‘ServerName’ directive globally to suppress this message
```

Go to the **/etc/apache2/apache2.conf** and insert the script below:

```
ServerName localhost
```

Reload the Apache, and the error will disappear, like in the image below:



Remove error AH00558

WARNING
You can change the localhost to your domain name, like website1.com or another domain name.

References

httpd.apache.org
phoenixnap.com
medium.com
digitalocean.com
stackoverflow.com
serverfault.com
baeldung.com
askubuntu.com

How to Storage Mount Using Bind Mount on Docker?

written by sysadmin | 26 June 2025

The previous article explained how to make storage using volume in Docker. This article will explain how to make storage using Bind Mount.

Problem

How to storage mount using bind mount on Docker?

Solution

A bind mount is a method of hosting a directory or file that is directly mounted into a container. Since they aren't isolated by Docker, both non-Docker processes on the host and container processes can modify the mounted files simultaneously. So, you can change the data from the host, and those changes will be reflected in the container. This method is useful for development environments where code must be updated and tested in real-time. There are 2 ways when use bind mount:

A. using -v option

This option (using **-v** or **--volume**) uses three fields, separated by colon characters (:). The fields must be in the correct order, and the meaning of each field is not immediately obvious. To use this option, use the format below:

```
docker run -d –name container_name -v
/path/folder/in/server:/path/folder/in/container[:opts]
image_name:tag
```

INFO

opts is the abbreviation for options like readonly, private, z, Z, and so on. The third field is optional, and is separated by colon characters. You can see the options and their descriptions on this page.

To see more detailed information about mounting a container, use the format below:

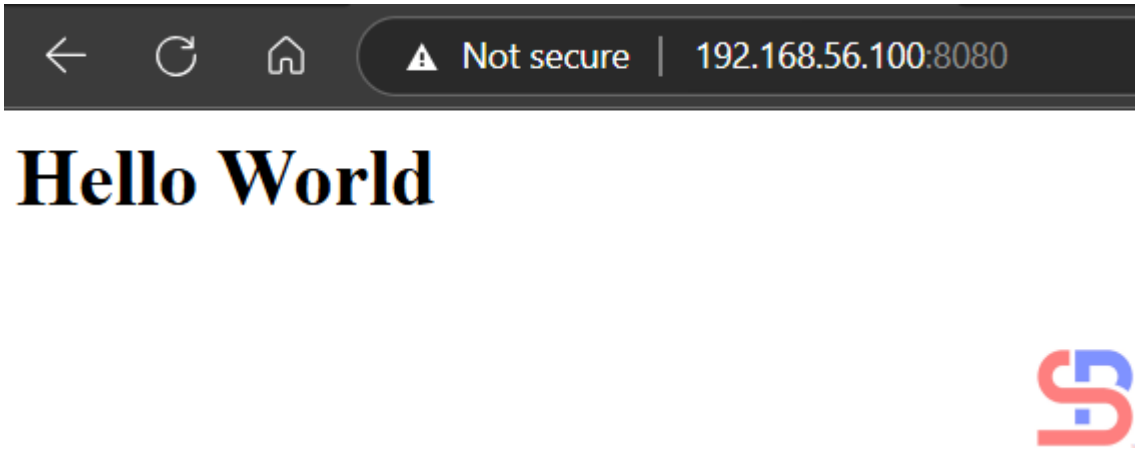
```
docker inspect container_name -f "{{json .Mounts}}" | python3 -m json.tool
```

For example, you want to create a container with a nginx webserver that can be accessed with port **8080** with sources in folder **/home/sysadmin/web** and destination to the folder **/usr/share/nginx/html** on container, so first make a web folder in the server, and create a simple site then create a container using the command below:

```
mkdir /home/sysadmin/web
echo "<h1>Hello World</h1>" > web/index.html
docker run -d --name webapp1 -p 8080:80 -v
/home/sysadmin/web:/usr/share/nginx/html nginx
docker inspect webapp1 -f "{{json .Mounts}}" | python3 -m json.tool
docker exec webapp1 cat /usr/share/nginx/html/index.html
```

Execute the commands

You can see from the image above that the index.html file in the container only displays the hello world sentence, so when you open the browser then the displays in the browser as shown below:



The display of the website

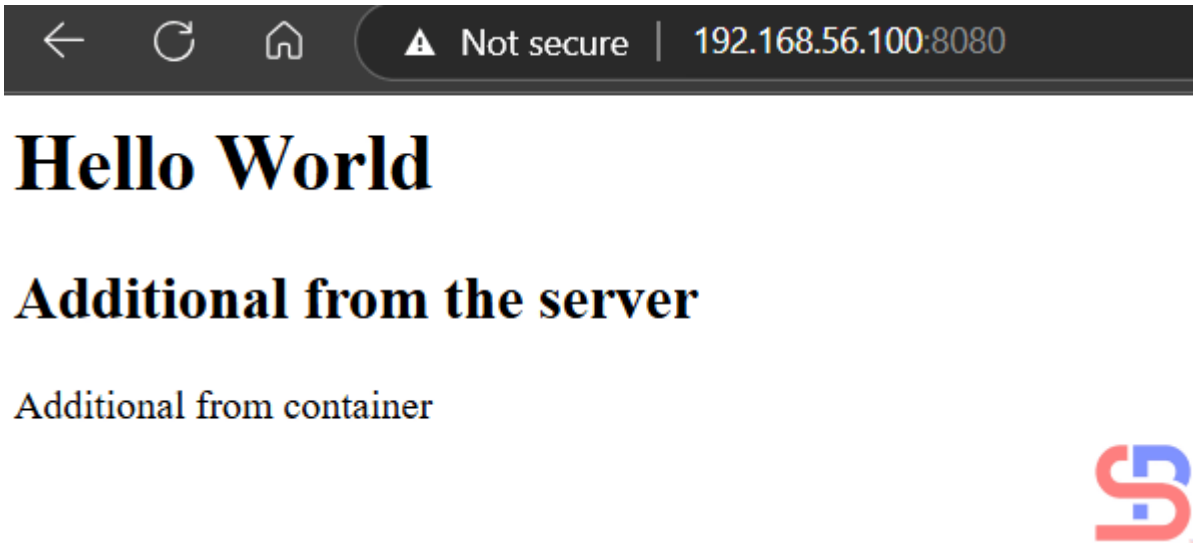
You can change the appearance of the website in the index.html file on the server or in the container. For example, you add the script below to the file on the server:

```
echo "<h2>Additional from server</h2>" >> web/index.html
```

Likewise, if you add the script below to the container:

```
docker exec webapp1 bash -c "echo Additional from container >>
/usr/share/nginx/html/index.html"
```

Both scripts will be included in the index.html file and will be displayed on the website as shown below:



Display of the website after additional scripts

B. Using --mount Option

In general, these options are more explicit and verbose. This option consists of multiple key-value pairs, separated by commas, and each consisting of a **<key>=<value>** tuple. The biggest difference is that the -v syntax combines all the options in one field, while the –mount syntax separates them. The –mount syntax is more verbose than -v or –volume, but the order of the keys is not significant, and it is easier to understand. To use this option, use the format below:

```
docker run -d --name container_name --mount
type=type_mount,source=/path/folder/in/server,destination=/path/folder/in/con
tainer[,<key>=<value>...] image_name:tag
```

INFO

The third field is optional, and is separated by commas like readonly, bind-propagation, and so on. You can see the options and their descriptions on this page.

For example, you want to create a container with an Apache web server that can be accessed on port **8081** with sources in the folder **/home/sysadmin/mount** and destination to the folder **/usr/local/apache2/htdocs** on the container, so first make the mount folder in the server, then make the container using the command below:

```
mkdir /home/sysadmin/mount
echo "<h1>Learn --mount option</h1>" > mount/index.html
docker run -d --name webapp2 -p 8081:80 --mount
type=bind,source=/home/sysadmin/mount,destination=/usr/local/apache2/htdocs
httpd
docker inspect webapp2 -f "{{json .Mounts}}" | python3 -m json.tool
docker exec webapp2 cat /usr/local/apache2/htdocs/index.html
```

```
sysadmin@docker:~$ mkdir /home/sysadmin/mount
sysadmin@docker:~$ echo "<h1>Learn --mount option</h1>" > mount/index.html
sysadmin@docker:~$ docker run -d --name webapp2 -p 8081:80 --mount type=bind,source=/home/sysadmin/mount,destination=/usr/local/apache2/htdocs httpd
5376279999b798a6f70f4ced5de48b07e387202984327421936d682a0fdcbb35
sysadmin@docker:~$ docker inspect webapp2 -f "{{json .Mounts}}" | python3 -m json.tool
[
    {
        "Type": "bind",
        "Source": "/home/sysadmin/mount",
        "Destination": "/usr/local/apache2/htdocs",
        "Mode": "",
        "RW": true,
        "Propagation": "rprivate"
    }
]
sysadmin@docker:~$ docker exec webapp2 cat /usr/local/apache2/htdocs/index.html
<h1>Learn --mount option</h1>
sysadmin@docker:~$
```

Execute the commands

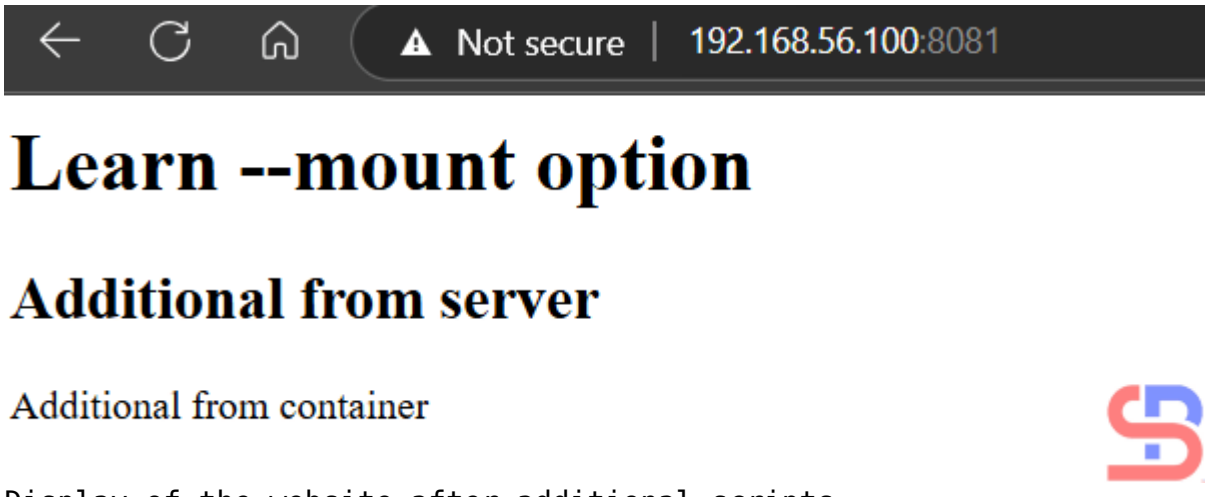
Like using the **-v** option, you can change the appearance of the website in the index.html file on the server or in the container. For example, you add the script below to the file on the server:

```
echo "<h2>Additional from server</h2>" >> mount/index.html
```

Likewise, if you add the script below to the container:

```
docker exec webapp2 bash -c "echo Additional from container >> /usr/local/apache2/htdocs/index.html"
```

Both scripts will be included in the index.html file and will be displayed on the website as shown below:



Display of the website after additional scripts

Note

The only difference between the two options above is how the commands are executed; otherwise, the results would be identical.

References

youtube.dimas-maryanto.com
youtube.com
docs.docker.com
docker.com
linkedin.com

How to Install and Configure NFS on Linux?

written by sysadmin | 26 June 2025

NFS or Network File Sharing is a protocol that allows you to share directories and files with other Linux clients over a network. Similar to locally created folders, an NFS file share is accessible when mounted on a client computer. When you have limited disk space and need to share public data between client machines, NFS is especially helpful.

Problem

How to install and configure NFS on Linux?

Solution

This article will explain how to install and configure NFS on 3 Linux distros: Rockylinux, Ubuntu, and OpenSuse and this article should work in each of their derivatives of the three distros.

A. On the server

Following are the steps to install and configure NFS:

1. Install NFS

I install NFS in the server with IP 192.168.56.2, and to install the NFS application on the Linux server, use the command below:

RockyLinux

```
sudo dnf install -y nfs-utils
```

Ubuntu

```
sudo apt update -y
sudo apt-get install -y nfs-kernel-server
```

OpenSUSE

```
sudo zypper install -y nfs-kernel-server nfs-utils
```

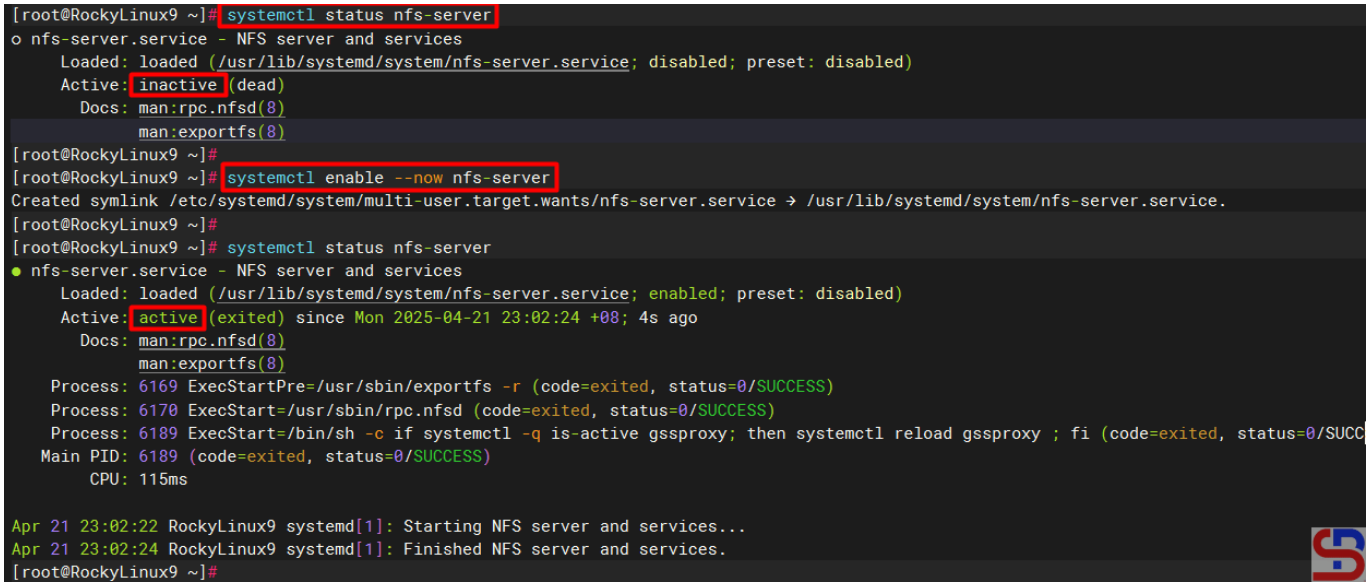
2. Check NFS status

Type the command below to check the NFS status:

```
systemctl status nfs-server
```

If you see the NFS status is still not on, use the command below to turn on the NFS service:

```
sudo systemctl enable --now nfs-server
```



Check the NFS service status

Sometimes you have to check the **nfs-mountd** service using the command below:

```
sudo systemctl status nfs-mountd
```

If the service is not on the server, then use the command below to turn on the service:

```
sudo systemctl start nfs-mountd
```

3. Check the rpcbind status

Make sure that the **rpcbind** service is actively used by NFS for the mapping port. Use the command below to check the status of the service:

```
sudo systemctl status rpcbind
```

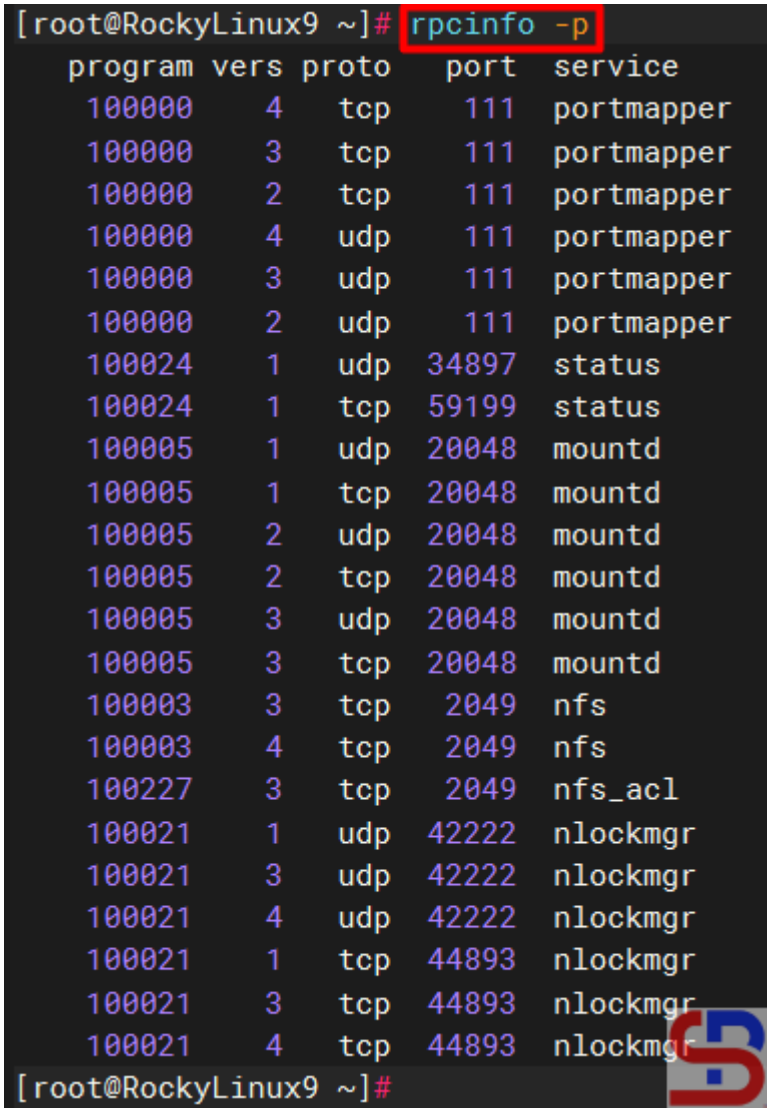
If the service is not active, use the command below to start the service:

```
sudo systemctl enable -now rpcbind
```

4. Check NFS and Portmap

To see if NFS and portmap (Portmap is a server that converts RPC program numbers into DARPA protocol port numbers. It must be running to make RPC calls) are running on the server, use the command below:

```
sudo rpcinfo -p
```



Check whether NFS and portmap run in the server or not

5. Configure firewall

If you still turn on the firewall on Linux, use the command below to open the NFS port (Port NFS is TCP Port 2049):

RockyLinux & OpenSUSE

```
firewall-cmd --add-service nfs --permanent
```

```
firewall-cmd --reload
firewall-cmd --list-services
```

```
[root@RockyLinux9 ~]# firewall-cmd --add-service nfs --permanent
success
[root@RockyLinux9 ~]#
[root@RockyLinux9 ~]# firewall-cmd --reload
success
[root@RockyLinux9 ~]#
[root@RockyLinux9 ~]# firewall-cmd --list-services
cockpit dhcpv6-client nfs ssh
[root@RockyLinux9 ~]#
```

Open the NFS port in RockyLinux

Ubuntu

```
sudo ufw allow nfs
sudo ufw status verbose
```

Use the command below to open the rpcbind port (rpcbind port is TCP Port 111):

Rockylinux & OpenSUSE

```
firewall-cmd --add-port=111/tcp --permanent
firewall-cmd --reload
firewall-cmd --list-ports
```

Ubuntu

```
sudo ufw allow 111
sudo ufw status verbose
```

6. Make a folder sharing

Create a folder to collect NFS files and folders and I make it in the folder **/var/nfs** using the command below:

```
mkdir /var/nfs
```

After that, copy the file(s) and folder(s) that you want to

share into the folder as shown below:

```
[root@RockyLinux9 ~]# mkdir /var/nfs
[root@RockyLinux9 ~]#
[root@RockyLinux9 ~]# cp -R * /var/nfs/
[root@RockyLinux9 ~]#
[root@RockyLinux9 ~]# ls -al /var/nfs
total 544
drwxr-xr-x.  3 root root    103 Apr 22 03:17 .
drwxr-xr-x. 20 root root   4096 Apr 22 03:17 ..
-rw-------.  1 root root   1321 Apr 22 03:17 anaconda-ks.cfg
-rw-r--r--.  1 root root  79793 Apr 22 03:17 download.htm
-rw-r--r--.  1 root root 103917 Apr 22 03:17 image.jpeg
-rw-r--r--.  1 root root 358300 Apr 22 03:17 quickmail.zip
drwxr-xr-x.  3 root root     21 Apr 22 03:17 uploads
[root@RockyLinux9 ~]#
```

Copy the file(s) and folder(s) into the folder sharing

7. Define an Export File

To grant access to NFS clients, you need to define an export file and it is typically located at **/etc/exports**. Use the format below to define an export file:

```
/folder/path    accessible-host-ip-address(options)
```

The options you can use can be seen in the image below:

Option	Description
ro,rw	Read-only, Read-write (default)
rw=list	Hosts in the list can do rw, others ro only
root_squash	Maps UID 0 and GID 0 to the value of anonuid and anongid (default)
no_root_squash	Allow root access
all_squash	Maps all UID and GID to anonymous one
subtree_check	Check that the accessed file is in the appropriate filesystem and in the exported tree.
no_subtree_check	Disables subtree checking
anonuid=xxx	Related to root_squash
anongid=xxx	Related to root_squash
secure	Require remote access from privileged port
insecure	Allow remote access from any port
noaccess	Prevent access to this dir and it's subdir

Options in NFS (Image credit for slideplayer.com)

You can use more than one option like (rw,sync,no_subtree_check). By default, NFS uses the **ro** option where the client can only read the file or folder in the folder sharing. In this article, I only want the folder sharing can only be accessed by users who only use IP 192.168.56.0/24 and the folder can be changed by the users, then use the command below to enter the script into the exports file:

```
sudo echo "/var/nfs                192.168.56.0/24(rw)" > /etc/exports
```

Then change the permissions so that the files and folders in the folder sharing can be changed using the command below:

RockyLinux & OpenSUSE

```
chown -R nobody:nobody /var/nfs
sudo chmod -R 775 /var/nfs
```

Ubuntu

```
chown -R nobody:nogroup /var/nfs
sudo chmod -R 775 /var/nfs
```

8. Export exports file

Use the command below to make the folder sharing available to the clients:

```
sudo exportfs -r
```

Use the command below to view the exports file:

```
showmount -e
```

To see which hosts access file sharing, use the command below:

```
sudo netstat -an | grep 2049
```

B. On the client

Following are the steps to install and configure NFS:

1. Install NFS client

Use the command below to install the NFS client:

RockyLinux

```
sudo dnf install -y nfs-utils
```

Ubuntu

```
sudo apt-get install -y nfs-common
```

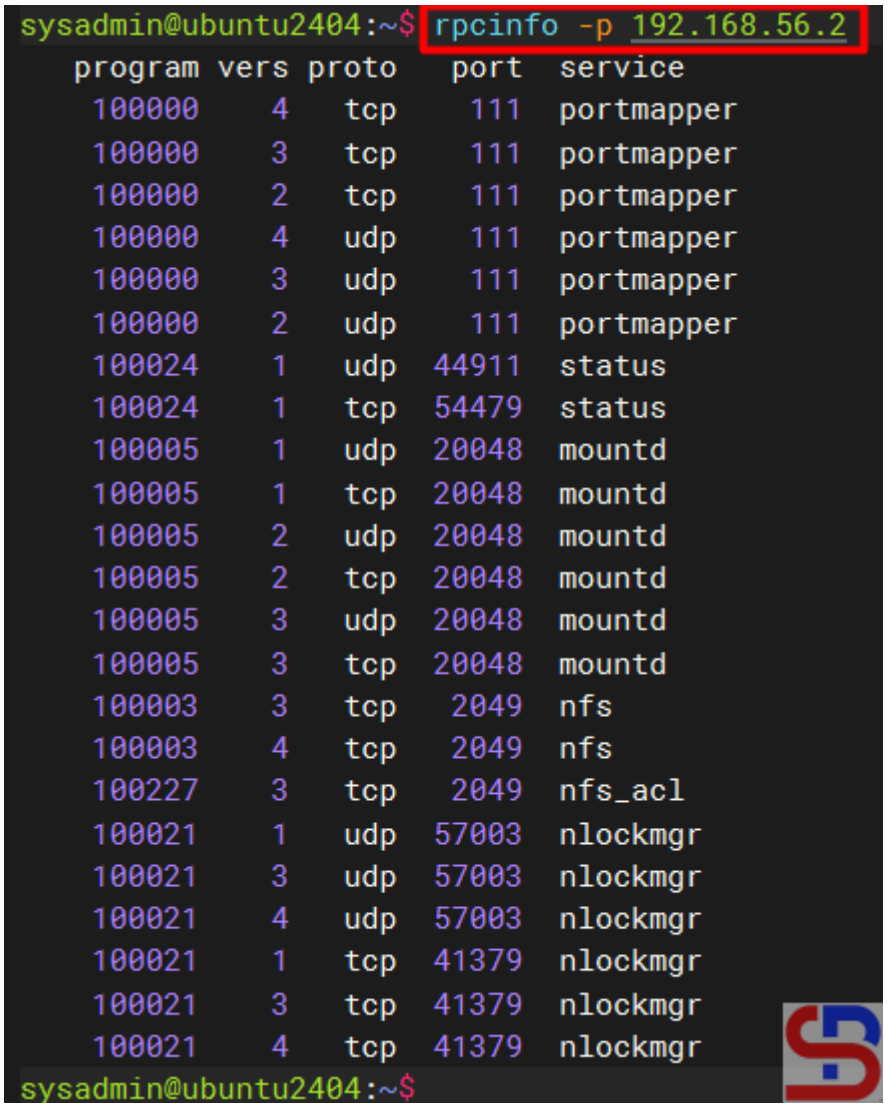
OpenSUSE

```
zypper install -y nfs-client*
```

2. Check the ports in the NFS server

Use the command below to check whether the client can access the ports (port 2049 and 111) in the NFS server or not (the IP server NFS is 192.168.56.2):

```
rpcinfo -p 192.168.56.2
```



Check the connection between the client to the NFS server

2. Make and mount a folder

Make the folder where we want to mount the NFS shares from the server, for example, I made a folder in **/tmp/nfs**:

```
mkdir /tmp/nfs
```

After that the mount folder with the NFS server using the format below:

```
sudo mount -t nfs 192.168.56.2:/var/nfs /tmp/nfs
```

```
sysadmin@ubuntu2404:~$ mkdir /tmp/nfs
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ sudo mount -t nfs 192.168.56.2:/var/nfs /tmp/nfs
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ df -h
Filesystem                         Size  Used Avail Use% Mounted on
tmpfs                               97M  1.1M   96M   2% /run
/dev/mapper/ubuntu--vg-ubuntu--lv  9.8G  4.8G  4.6G  52% /
tmpfs                              481M     0  481M   0% /dev/shm
tmpfs                              5.0M     0  5.0M   0% /run/lock
/dev/sda2                          1.7G   95M  1.5G   6% /boot
tmpfs                               97M   12K   97M   1% /run/user/1000
192.168.56.2:/var/nfs               17G  1.6G   16G  10% /tmp/nfs
sysadmin@ubuntu2404:~$
```

Mount the folder to the folder-sharing

INFO

You can use the **-v** option so that the above command becomes:
sudo mount -v -t nfs 192.168.56.2:/var/nfs /tmp/nfs
to display the logs when mounting so that you can know if there is an error when mounting.

You should access the folder sharing on the NFS server as shown below:

```
sysadmin@ubuntu2404:~$ sudo mount -t nfs 192.168.56.2:/var/nfs /tmp/nfs
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ df -h
Filesystem                         Size  Used Avail Use% Mounted on
tmpfs                               97M  1.1M   96M   2% /run
/dev/mapper/ubuntu--vg-ubuntu--lv  9.8G  4.3G  5.0G  47% /
tmpfs                              481M     0  481M   0% /dev/shm
tmpfs                              5.0M     0  5.0M   0% /run/lock
/dev/sda2                          1.7G   95M  1.5G   6% /boot
tmpfs                               97M   12K   97M   1% /run/user/1000
192.168.56.2:/var/nfs               17G  1.7G   16G  10% /tmp/nfs
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ ls /tmp/nfs
anaconda-ks.cfg  download.htm  image.jpg  quickmail.zip  uploads
sysadmin@ubuntu2404:~$
```

Access to the NFS server

You can use the command below to see the NFS client connection:

```
sudo mount | grep -i nfs
```

```
sysadmin@ubuntu2404:~$ sudo mount | grep -i nfs
192.168.56.2:/var/nfs on /tmp/nfs type nfs4 (rw,relatime,vers=4.2,rsize=131072,wsize=131072,namlen=255,hard,proto=tcp,timeo=600,retrans=2,sec=sys,clientaddr=192.168.56.100,local_lock=none,addr=192.168.56.2)
sysadmin@ubuntu2404:~$
```

Check the status of the NFS client

4. Simulation test

Try to do the simulation by changing the file name in the folder sharing. I try to rename the download.htm file to index.html using the command below:

```
sudo mv /tmp/nfs/download.htm /tmp/nfs/index.html
```

The file was successfully changed as shown below:

```
sysadmin@ubuntu2404:~$ ls /tmp/nfs
anaconda-ks.cfg  download.htm  image.jpeg  quickmail.zip  uploads
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ sudo mv /tmp/nfs/download.htm /tmp/nfs/index.html
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ ls /tmp/nfs
anaconda-ks.cfg  image.jpeg  index.html  quickmail.zip  uploads
sysadmin@ubuntu2404:~$
```

Rename the file in NFS

5. Configure the fstab file

To keep the folder sharing is still connected in the client after the client is rebooted, configure the **/etc/fstab** file using the command below:

```
echo '192.168.56.2:/var/nfs        /tmp/nfs        nfs        rw 0 0' | sudo tee -a /etc/fstab
```

```
[root@RockyLinux9 ~]# df -h
Filesystem                      Size  Used Avail Use% Mounted on
devtmpfs                        4.0M     0  4.0M   0% /dev
tmpfs                           385M     0  385M   0% /dev/shm
tmpfs                           154M  3.1M  151M   2% /run
/dev/mapper/rl_rockylinux9-root  17G  1.8G   16G  11% /
/dev/sda1                      1014M  395M  620M  39% /boot
192.168.56.12:/var/nfs           10G  3.4G  5.8G  37% /tmp/nfs
tmpfs                            77M     0   77M   0% /run/user/1000
[root@RockyLinux9 ~]#
[root@RockyLinux9 ~]# sudo echo '192.168.56.12:/var/nfs /tmp/nfs        nfs     rw 0 0' | sudo tee -a /etc/fstab
192.168.56.12:/var/nfs /tmp/nfs        nfs     rw 0 0
[root@RockyLinux9 ~]#
[root@RockyLinux9 ~]# cat /etc/fstab

#
# /etc/fstab
# Created by anaconda on Thu Sep 19 07:29:32 2024
#
# Accessible filesystems, by reference, are maintained under '/dev/disk/'.
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.
#
/dev/mapper/rl_rockylinux9-root /                       xfs     defaults        0 0
UUID=066eb699-fd9c-45ae-bba8-6c220e767ed7 /boot                   xfs     defaults        0 0
/dev/mapper/rl_rockylinux9-swap none                    swap    defaults        0 0
192.168.56.12:/var/nfs /tmp/nfs        nfs     rw 0 0
[root@RockyLinux9 ~]#
```

Insert the script to fstab file

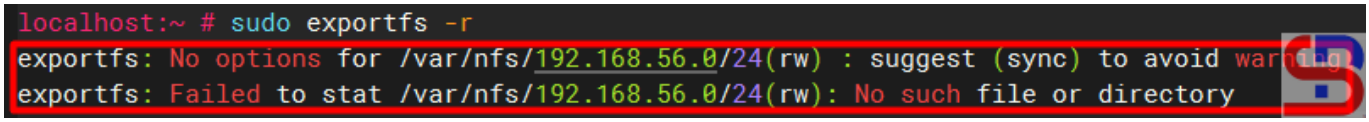
C. Errors and solutions

Below are errors that often appear and their solutions:

1. No options for /var/nfs

Sometimes when you run the **exportfs -r** command, there is an error as below:

```
exportfs: No options for /var/nfs/192.168.56.0/24(rw) : suggest (sync) to avoid warning
exportfs: Failed to stat /var/nfs/192.168.56.0/24(rw): No such file or directory
```



Error failed to stat

To eliminate the error, check in the **/etc/exports** file and you have to fix the writing in the file from:

/var/nfs/192.168.56.0/24(rw)

changed into

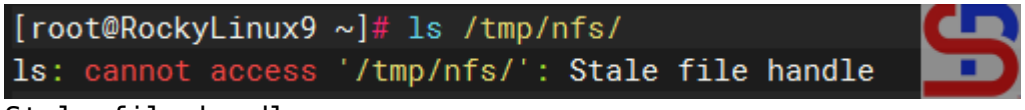
/var/nfs 192.168.56.0/24(rw)

After that, run the **exportfs -r** command again and the error should disappear.

2. Error Stale file handle

When you want to connect a client to the NFS server there is an error like the below (usually this happens if there is an error like number 1 or other causes on the NFS server):

```
Stale file handle
```



Stale file handle error

To solve this error you have to unmount on the side of the client and then mount back as shown below:

```
[root@RockyLinux9 ~]# ls /tmp/nfs/
ls: cannot access '/tmp/nfs/': Stale file handle
[root@RockyLinux9 ~]# sudo umount -f /tmp/nfs
[root@RockyLinux9 ~]# df -h
Filesystem                         Size  Used Avail Use% Mounted on
devtmpfs                           4.0M     0  4.0M   0% /dev
tmpfs                              385M     0  385M   0% /dev/shm
tmpfs                              154M  3.1M  151M   2% /run
/dev/mapper/rl_rockylinux9-root     17G  1.8G   16G  11% /
/dev/sda1                         1014M  395M  620M  39% /boot
tmpfs                               77M     0   77M   0% /run/user/1000
[root@RockyLinux9 ~]#
[root@RockyLinux9 ~]# sudo mount -t nfs 192.168.56.12:/var/nfs /tmp/nfs
[root@RockyLinux9 ~]# df -h
Filesystem                         Size  Used Avail Use% Mounted on
devtmpfs                           4.0M     0  4.0M   0% /dev
tmpfs                              385M     0  385M   0% /dev/shm
tmpfs                              154M  3.1M  151M   2% /run
/dev/mapper/rl_rockylinux9-root     17G  1.8G   16G  11% /
/dev/sda1                         1014M  395M  620M  39% /boot
tmpfs                               77M     0   77M   0% /run/user/1000
192.168.56.12:/var/nfs              10G  3.4G  5.8G  37% /tmp/nfs
[root@RockyLinux9 ~]# ls /tmp/nfs/
bin  get-docker.sh
[root@RockyLinux9 ~]#
```

Solve the stale file handle error

3. RPC: Program not registered

When typing the **showmount -e** command on the NFS server there is an error as below:

```
clnt_create: RPC: Program not registered
```

```
localhost:~ # showmount -e
clnt_create: RPC: Program not registered
```

Error Program Not Registered

The solution is that you have to run the command below so that the nfs-mountd service runs on the server:

```
systemctl start nfs-mountd
```

4. Permission denied

When you want to connect to the NFS server or when you want to change the file in the NFS, there is an error like this:

Permission denied

```
sysadmin@ubuntu2404:~$ cp /tmp/nfs/anaconda-ks.cfg /tmp/nfs/test.cfg
cp: cannot create regular file '/tmp/nfs/test.cfg': Permission denied
```

Error Permission denied

The solution is to check the exports file on the NFS server and make sure that the folder has been given permissions as in step 5 in the server section.

Note

If you want to block an IP address of a host so the host can't access the NFS server, use the command below to block the IP host:

```
sudo firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source
address="192.168.56.100" port port="2049" protocol="tcp" reject'
sudo firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source
address="192.168.56.100" port port="111" protocol="tcp" reject'
sudo firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source
address="192.168.56.100" port port="2049" protocol="udp" reject'
sudo firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source
address="192.168.56.100" port port="111" protocol="udp" reject'
sudo firewall-cmd --reload
sudo firewall-cmd --list-rich-rules
```

and should the client with IP 192.168.56.100 not be able to access the folder sharing as shown in the image below:

```
sysadmin@ubuntu2404:~$ ip a | grep 56
    inet 192.168.56.100/24 brd 192.168.56.255 scope global enp0s8
sysadmin@ubuntu2404:~$
sysadmin@ubuntu2404:~$ sudo mount -t nfs 192.168.56.2:/var/nfs /tmp/nfs
mount.nfs: Connection refused for 192.168.56.2:/var/nfs on /tmp/nfs
sysadmin@ubuntu2404:~$
```

Can not mount to NFS server

If you want to delete an IP address of a host then the option **--add-rich-rule** becomes **--remove-rich-rule** so that the command becomes as command below:

```
sudo firewall-cmd --permanent --remove-rich-rule='rule family="ipv4" source
address="192.168.56.100" port port="2049" protocol="tcp" reject'
sudo firewall-cmd --permanent --remove-rich-rule='rule family="ipv4" source
address="192.168.56.100" port port="111" protocol="tcp" reject'
sudo firewall-cmd --permanent --remove-rich-rule='rule family="ipv4" source
address="192.168.56.100" port port="2049" protocol="udp" reject'
sudo firewall-cmd --permanent --remove-rich-rule='rule family="ipv4" source
address="192.168.56.100" port port="111" protocol="udp" reject'
sudo firewall-cmd --reload
sudo firewall-cmd --list-rich-rules
```

WARNING
In my experience, you can’t immediately block a client to NFS if the client is still connected to the NFS. You have to wait until the client disconnects to the NFS server, either the host reboots or others.

References

bluexp.netapp.com
redhat.com
phoenixnap.com
howtoforge.com
youtube.com
docs.oracle.com
linux.die.net