Implementing secure access for NFS server

In this example, we will be implementing wireguard vpn to securely access our NFS server VM. Our NFS will only connection from the VPN subnet and also users will require VPN credentials.

First as usual run sudo apt update && sudo apt upgrade -y

Next install wireguard with sudo apt install wireguard

A screenshot of a computer

Description automatically generated

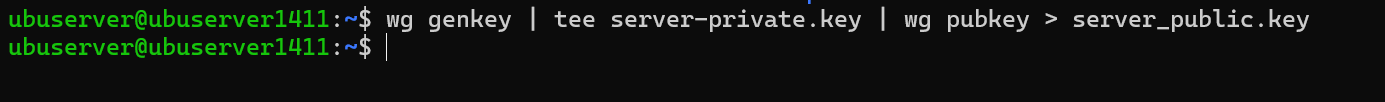
Now we will need to configure our wireguard server

Run umask 077 so that only owner can read/write the files being created

A black and white image of a calculator

Description automatically generated

Now run the following to generate a private key and save it to server-private.key and server\_public.key.



If you want to know where the keys are stored its in your home directory or you can use locate server-private.key (need install plocate first). Now if you do a ls -l you can see the only rw permission is enabled for your user which is correct as we used umask 077 earlier.

A screenshot of a computer

Description automatically generated

Now that the keys are generated you can change it back to default with umask 022



For umask 077

666 - 077 = 600 (files)

777 - 077 = 700 (directories)

For umask 022

666 - 022 = 644 (files)

777 - 022 = 755 (directories)

Next, do a sudo vim /etc/wireguard/wg0.conf and paste in the following. Replace private key with what is shown from sudo cat ~/server-private.key

A black background with white text

Description automatically generated

What this config file means:

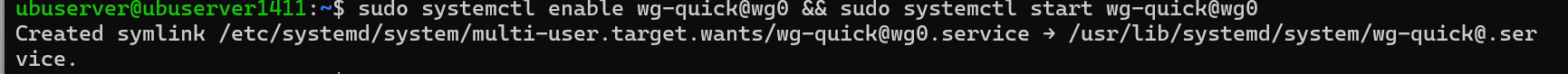
10.8.0.1/24 will be the server’s IP on the VPN network. This is not the same as the VM physical network interface 192.168.x.x/24. 10.8.0.1 is just for VPN clients to connect.

Save config = wireguard will save any runtime changes to config file

Listen port 51820 as it’s the default wireguard port and the UDP port wireguard will listen on

Next enable and start the wireguard interface

You can combine both with the && operator. Once it says created symlink means service was enabled successfully, you can also use systemctl status wg-quick@wg0 to verify. Also if you named your /etc/wireguard/ config file differently (e.g. vpn.conf) then you would need to use wg-quick@vpn instead of wg-quick@wg0



You can also use sudo wg show to see if the interface is up. It should display your public key as well as the listening port.

A green numbers on a black background

Description automatically generated

Use sudo ufw allow 51820/udp to add the wireguard port to firewall list

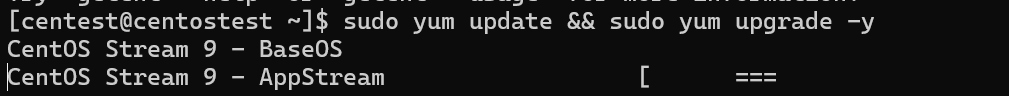
A black background with white text and colorful symbols

Description automatically generated

Now that wireguard is setup on your server, we now have to setup for each client

In this example, we ssh into our centos vm client

[CLIENT]:As usual update packages first

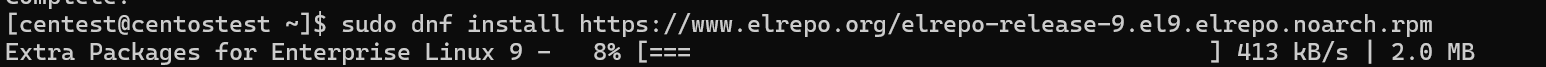


[CLIENT]:For CentOS, install epel-release first

A screen shot of a computer

Description automatically generated

[CLIENT]:Then add elrepo

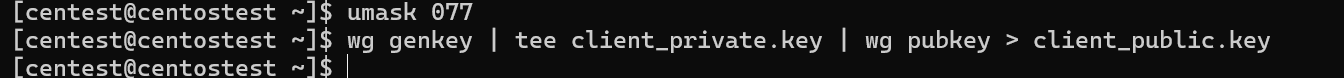


[CLIENT]:Then run sudo yum install wireguard-tools

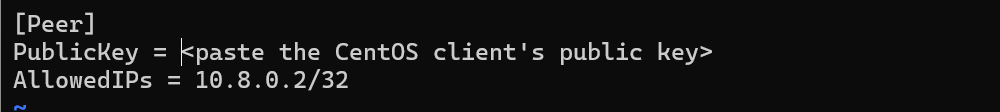
A screenshot of a computer

Description automatically generated

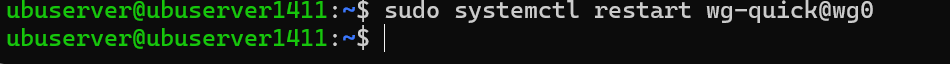
[CLIENT]:Similar to how we setup for the server, we generate the client private and public key now. Remember to set back umask to 022 for default.



[SERVER]:Now, add a section at the bottom of your server’s wg0.conf file and paste the generated client public key inside



[SERVER]:Apply changes with sudo wg set wg0 <add the client public key> allowed-ips 10.8.0.2/32

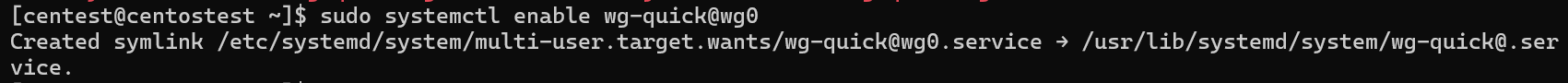


[CLIENT]: Now do a sudo vim /etc/wireguard/wg0.conf and add the following in:

A screenshot of a computer screen

Description automatically generated

[CLIENT]: Enable and start wireguard service





[CLIENT]: to verify all is working, you can use sudo wg show. You should also see another section at the bottom reflecting your server’s public key.

A black background with white text

Description automatically generated

[CLIENT]: now you should be able to ping to your server

A screen shot of a computer screen

Description automatically generated

[SERVER]: and from server to client

A black screen with white text

Description automatically generated

Now that we have settled basic VPN connection and testing we can proceed to restrict access to VPN clients only for our NFS share.

Change your /etc/exports file to only allow connection from the VPN subnet



Apply chages



To test if you can still mount your NFS server share without VPN, you can disconnect centest from VPN and test mount /srv/nfs/shared again

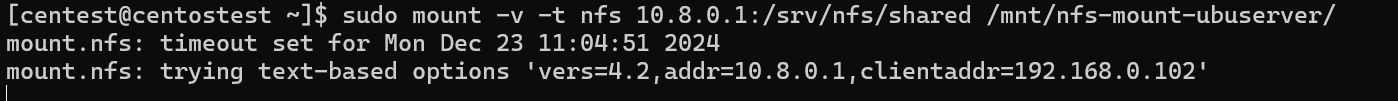


Using normal method mounting direct to server ip:

A black screen with white text

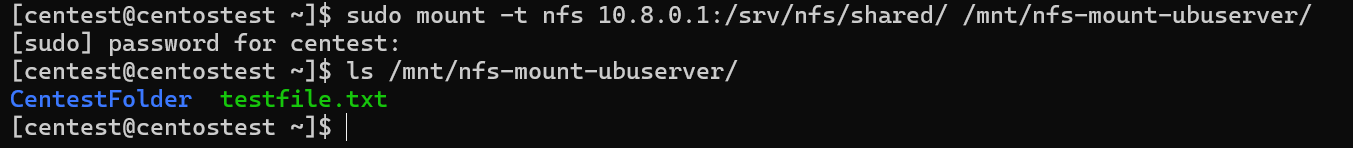
Description automatically generated

If you try to mount to server VPN interface (wg0) without running wireguard it will timeout:



Hence, we have now restricted connection to our mount point via the exports config which can only be connected if wireguard is installed and have peer connection to our server.

However once you enable back wg0 interface you should be able to mount it already



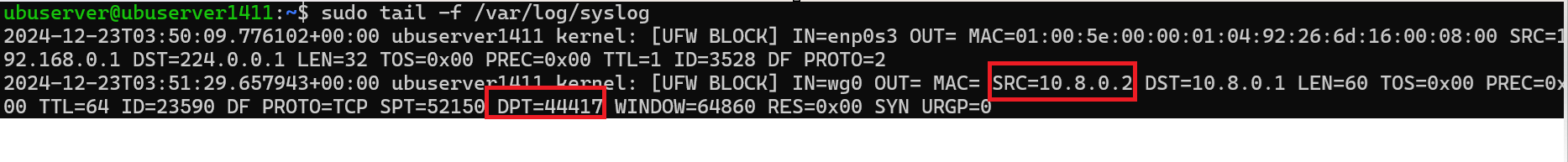
Side note that if your vpn interface is up you should be able to see wg0 in ifconfig as well

A black screen with white text

Description automatically generated

[SERVER]: If your client is not able to connect to the server you may try to check /var/log/syslog to see if it’s being blocked. You can force NFS to use specific port and then allow in firewall (ufw) for those ports you specify.

First you can do a showmount -e serverip from your client machine and on server machine do a tail -f /var/log/syslog to view what port the nfs service is using

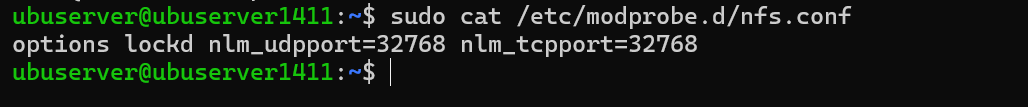


Then we can try set fixed port: edit the /etc/default/nfs-kernel-server by forcing mountd to use 20048 and quota service to use 32769. Mountd handles mount requests from clients and quota service limits how much disk space users/groups can use.

A computer screen shot of a black screen

Description automatically generated

Editing /etc/modprobe.d/nfs.conf to set tcp and udp port for file locking



Once done, next add the following ufw rules

Sudo ufw allow from 10.8.0.0/24 proto tcp to any port 44417

Sudo ufw allow from 10.8.0.0/24 proto udp to any port 44417

Sudo ufw allow from 10.8.0.0/24 proto tcp to any port 32768

Sudo ufw allow from 10.8.0.0/24 proto udp to any port 32768

32768: lockd

44417: NFSv4 callback service

A screenshot of a computer program

Description automatically generated