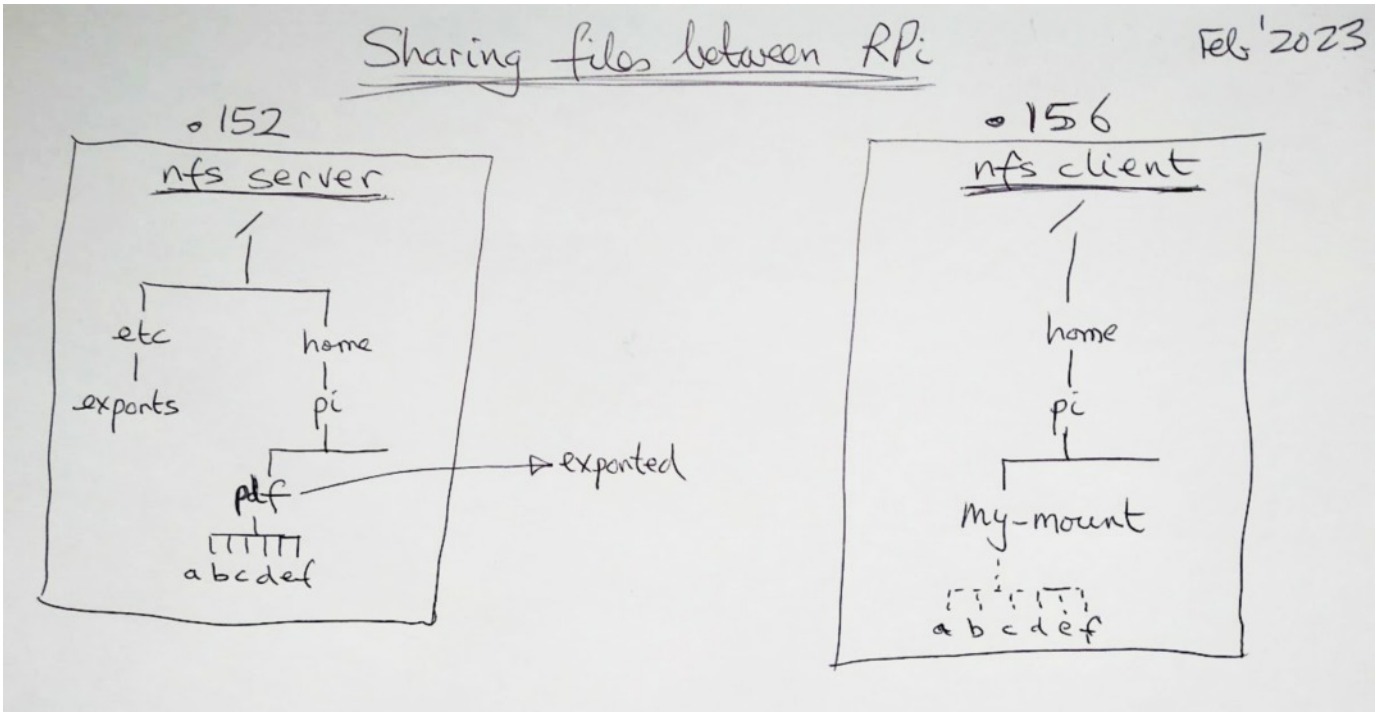
Bohunt School (Wokingham)

Sharing files between Raspberry Pi(es)

This step-by-step guide should enable you to set-up and share files between two or more Raspberry Pi devices on the same network using NFS.

Although the Network Filing System (NFS) was originally by Sun Microsystems in 1984, it is still very popular today and useful as data transmission medium. It is supported by a number of platforms and, as you will see, is very easy to setup.

Here's one of Mr D's famous freehand sketches. It shows two Raspberry Pi devices - one acting as a server and the other one as a client.



In this tutorial the Raspberry Pi with IP address 192.168.1.152 will act as a NFS server and will share the files located in the directory at… /home/pi/pdf

The Raspberry Pi with IP address 192.168.1.156 will act as a NFS client and will 'mount' the exported file structure onto a local directory - which in this example is located at… /home/pi/my_mount
If you entered the Raspberry Pi OS command 'ls' in this directory, you would see the files that are actually located in the remote 'pdf' directory.

So what's the benefit?

The main benefit is convenience and ease of access. For example, maybe one of your colleagues has created a Node-RED flow to obtain a weather report from OpenWeatherMap or some other provider. That data could be made available to you and your Node-RED flow by sharing files using NFS.

It should be noted there are other ways of sharing data between two systems (e.g. File Transfer Protocol - FTP) which will be discussed in another tutorial.

Bohunt School (Wokingham)

Sharing files between Raspberry Pi(es)

OK - let's get started in setting up our two Raspberry Pi devices.

First of all let's bring both of the Raspberry Pi devices up-to-date.

1. **Enter… sudo apt update in a Terminal window**
2. **Enter… sudo apt full-upgrade in a Terminal window**

Setting-up the NFS server (on 192.168.1.152)

1. **Create the directory to be shared (if it doesn't exist already).**
   In this example it is called 'pdf' but it could have been called 'shared_files' or any other name of your choosing.

   mkdir -p /home/pi/pdf

2. **Install the NFS server program.**
   sudo apt-get install nfs-kernel-server

3. **Edit the table that defines what will be exported.**
   sudo nano /etc/exports

Add this line to the file…

/home/pi/pdf *(rw,sync,no_subtree_check)

Save this file by clicking Ctrl+x and then Y

4. **Run this command to export the updated table.**

   sudo exportfs -a

5. **Use this command to restart the NFS server**

   sudo systemctl restart nfs-kernel-server

6. **At this stage you can check what is being exported using this command.**

   sudo exportfs

Setting-up the NFS client (on 192.168.1.156)

1. **Install the NFS client program**
   sudo apt-get install nfs-common

2. **Use this command to create a mount point.**
   mkdir -p /home/pi/my_mount

3. **Mount the remotely shared folder on this mount point.**
   sudo mount -t nfs 192.168.1.152:/home/pi/pdf /home/pi/my_mount

Checking to see if it has worked

1. In a terminal window enter… cd /home/pi/my_mount
2. Enter the 'ls' command in a terminal window - you should see the names of the files (that are actually located on the remote server).

```
pi@iot-super-server:~/my_mount $ ls
david.pdf  newsletter_demo.pdf
pi@iot-super-server:~/my_mount $
```

This image shows the files that are being exported on the remote machine.

```
pi@fun-super-server:~/pdf $ ls
david.pdf  newsletter_demo.pdf
pi@fun-super-server:~/pdf $
```

As you can see the two listings are exactly the same.

The NFS share you have created is Read as well as Write, this means if you save a new file in… /home/pi/my_mount it will appear in… /home/pi/pdf on the remote server.

As an example, a new file has been stored in the local machine.

```
pi@iot-super-server:~/my_mount $ ls
david.pdf  newsletter_demo.pdf  this_is_a_new_file.txt
pi@iot-super-server:~/my_mount $
```

As you can see, it appears in the remote machine.

```
pi@fun-super-server:~/pdf $ ls
david.pdf  newsletter_demo.pdf  this_is_a_new_file.txt
pi@fun-super-server:~/pdf $
```

Un-mounting a filing system

Just as you can 'mount' a filing system, you can also un-mount it (or disconnect the link) with the 'umount' command.

So to disconnect the connection you made, you could enter this command in a Terminal window on the local machine.

```
sudo umount /home/pi/my_mount
```

Quite probably the local machine may be using one or more of the (exported) files or the Terminal window may still be located in the 'my_mount' directory.

This will probably result in the following error message being generated.

```
pi@iot-super-server:~/my_mount $ sudo umount /home/pi/my_mount
umount.nfs4: /home/pi/my_mount: device is busy
pi@iot-super-server:~/my_mount $
```

The -l option can be used to force the un-mounting task to be executed.

sudo umount /home/pi/my_mount -l

Making things permanent

The mount command you entered, on the previous page, was performed manually. This means if the Raspberry Pi is rebooted or suffered a power outage, the remotely shared directory would not be mounted anymore.

You can overcome this situation in two ways.

1. Issue a 'mount' command manually by entering it from a terminal window or using an 'exec' node on a Node-RED flow to perform the same task.

2. Modify the contents of the 'fstab' file to perform this task at every boot-up.

Modify the contents of the 'fstab' file

It is a fairly simple task to modify this file. Just follow these steps.

1. Edit the 'fstab' file.
   sudo nano /etc/fstab

2. Insert a new line to define the filing system to be mounted at boot time.

   192.168.1.152:/home/pi/pdf /home/pi/my_mount nfs defaults,noauto,x-systemd.automount 0 0

   The first part denotes the file server and path to the shared directory

   The second part is the local mount point

   'nfs' defines the type of filing system, to be used

3. Save this file by clicking Ctrl+x and then Y

Next time your Raspberry Pi boots-up it should automatically mount the filing system you defined in the 'fstab'.

If you want to perform this task manually you can enter this command in a terminal window… sudo mount -all

You now know how to share files across a pair of Raspberry Pi devices on the same network. Have fun exploring what you can do with 'mount' command.

Congratulations - you are now able to shares files across your local network