

## How To Make A Dedicated File Server Or Web Server

Have an old computer just lying around in your garage doing nothing? Well, I do, and I have decided to share my experiences with you.

The best way to give life to your old computer is to turn it into a dedicated fileserver or web server. I have done it, but I will leave instructions on how you can do it too!

### Step One: Survey Old Computer

First of all, survey the specs of the old computer. How much ram does it have, what's the processor and speed, and how big is the hard drive? In older systems, these are the main things that will determine its usefulness. For this article, I have used two different machines with these specs:

#### Machine One: (Vintage)

- Intel Pentium 75mhz
- 48mb EDO Ram
- 2x4.1gb Hard Drives
- 16x CDROM
- 10/100mbps Network Card

#### Machine Two: (Not So Bad)

- IBM Cyrix MIII 333mhz
- 128mb SDRAM
- 40gb Harc Drive
- 52x CDROM
- 10/100mbps Network Card

Now, looking at these specs, you may be mistaken by thinking that they are nothing but giant paper weights. Well, they aren't. I'll show you how to turn them into servers!

### Step Two: Get The Software

For my tutorial, I have decided to use a brand spanking new copy of Fedora Core 3. Yes, I am going to run Fedora Core 3 on these systems. You can download the whole set of Fedora Core cds from [fedora.redhat.com](http://fedora.redhat.com)

The requirements indicate that these systems can't run it, but don't worry, they will, if they are not running an X server.

### Step Three: Install

Once you have the cds all burned, just insert them into the computer at bootup. If you can't boot cds, then you need to use fedora core 1, which allows you to make boot floppies. Since we want these computers to do nothing but serve files and pages, we will not install an X server or window manager. If that worries you newbs out there, don't worry, I'll show you an easy way to configure the servers later in the article. Be mindful or what I.P address you assign to the server, make sure there are no conflicts with other I.P addresses on the network. Here is what I chose for my I.P Addresses.

Server: 192.168.1.1

Client Machine: 192.168.1.2, and so on...

When you install, make sure that you only install the bare essentials. I recommend that you install emacs, all of the server sections, and the redhat configs. Oh, and for systems that have tiny amounts of ram, such as the ones I used, make sure you have a heap of swap space. Once everything is done, it's off to restart them computer. Oh, and don't forget to make a floppy bootdisk!!!!

### Step Four: Post Configuration

If you are running a very slow machine (such as a pentium 1), you'll notice that it may take a while for Fedora to boot. My oldest system just spat out random numbers when it tried to boot grub, so I was forced to use a bootdisk everytime I needed to boot the computer, NOT FUN! Anyway, there is an easy way to make Fedora boot faster. Once the computer has booted and you are left with the user login, login as root. Now, in order to get Fedora to run faster, we need to cut down on all of the services that Fedora starts up with by default. Go to your client machine. Open up a console, and ssh into the server by typing in the following:

```
su
```

```
ssh -X 192.168.1.1
```

It will then give you a warning about key verification. Just say yes and it will ask you for the root password for the server. Just type it in and proceed. Now, the ssh command that we typed in allows us to run X on the client machine without bringing the server down to its knees. We need to get rid of some of the services at bootup, so on the client machine, type:

```
redhat-config-services
```

After a while, you'll see that the redhat-config-services program loads up. Just eliminate all of the services that you don't really need (kudzu, pcmcia, etc). Save the changes and quit the program. Next, we need to configure NFS, Samba, and Apache. It's really simple with Fedora. While you are still logged on to the server with ssh, you can configure these by the following commands:

```
redhat-config-samba  
redhat-config-nfs  
redhat-config-httpd
```

These will allow you to configure the appropriate services on the server via gui. Once these are set to your likings (I set samba and nfs to share the same folder), you can exit the ssh.

#### Step Five: Testing

OK, everything is done, all we need to do now is test it. If you want to share files with the server via linux, then using NFS is the best option. However, you'll have to tell your linux where everything is. To do that, you need to update your /etc/fstab. It's easy, just open up a console and login as root.

Then type :

```
emacs /etc/fstab
```

It will then open up your fstab. Now we need to write down our configuration. Go to the bottom of the file and make a new line. This is what I added to my fstab, if you are using the settings in this tutorial, then it will work for you, but you may add your own settings.

```
192.168.1.1:/share /share/ nfs wsize=8192,rsize=8192,timeo=14,intr
```

This is telling the client that there is a share file on the server with the i.p address of 192.168.1.1 and the folder that is to be shared in called share. It also tells the client where to mount the share, and what type of filesystem it is. You will now need to mount the share. You can do so by typing:

```
mount 192.168.1.1:/share /share/
```

It will pause for a moment and then mount the share. You can now browse to where you mounted it and write and read to it at your hearts content.

Now if you want to share files with the server with Windows, then you'll need to use the Samba protocol. Make sure you know what workgroup your samba is using, mine was "mygroup". You can change it if you like. Now, boot windows and then add a workgroup called "mygroup", or whatever your samba was called. It will then allow you to share with the server. You may need to reboot windows to get it working.

#### Step Six: Conclusion

Well, the old piece of junk is now doing something useful. You'll find that even an old pentium 75mhz can send files around at 5mbps, which isn't that bad unless you have a lot of clients. I found that the Cyrix machine that I used was a much better server simply because it really shot files through the network. I know I haven't touched on security matters like encryption and passwords, but if it is a private network, then you hardly need it, and I don't know crap about it. Perhaps you can add some comments about this, and give your own suggestions.

Not only does this tutorial tell you how to recycle old computers. but it also shows you the usefulness of the wonderful operating system called Linux

#### About the Author

server help, dedicated server, server tutorial, dedicated server guide, how to make a server, web servers, file servers, dedicated file server, dedicated

web server

Source: <http://www.citylinkpcs.com.au>