Creating a LAMP server (web server – Linux Apache Mysql PHP) with the Raspberry Pi

This provides details of how to configure a Raspberry Pi as a webserver. This is similar to the guide to using Xubuntu as a LAMP webserver, but adds some of the things that need to be handled differently for the Raspberry Pi.

It is termed a LAMP server which is one of the most common configuration for webservers which standard for:
Linux – operating system
Apache – webserver (http) software
Mysql – database server
PHP or **Perl** – programming languages

This setup is probably overkill for most uses of the Raspberry Pi, but it is the setup that most users will be familiar with and is a good way to learn about setting up a webserver. I'll be looking at setting up a lightweight setup in future.

All this configuration is done at the command line. This may not be quite as easy as clicking a few icons, but it has many advantages, including the ability to remotely manage and install the server. It also means that the computer can spend more of it's time server up web pages and less processor time drawing a GUI, which is after all the whole point of a webserver.

Why use the Pi?

Avoiding the geeking answer of "because I can" I think there are some good reasons for doing this.

Learn Linux The main aim of this site is to teach Linux skills. The best way to do that is to actually do something. This is as good a place as any to start.

Learn web programming The aim of the Raspberry Pi Foundation is to teach programming to children. Learning to program web based applications can be a useful skill to learn. It's debatable whether it's better to learn to program a desktop application before web programming or vice versa, but it's certainly something that is a useful skill to learn.

As an interface The Raspberry Pi is useful as a device for collecting information from various sensors. A web server can be a good way of accessing that information.

Dedicated network device You could use it as a dedicated network service for the home. Perhaps streaming videos / media etc. .

As a test or development server When creating a web application it is useful to have a dedicated server to test the application on. Ideally this should be identical hardware and software to the production, but if that is not possible then the Raspberry Pi could be an inexpensive alternative.

As a real web server At first I was going to put that it would be silly to run this as a production webserver. Then I thought back to the server I was running my personal blog website on, prior to going hosted in 2007, was not much more powerful than the Raspberry Pi is. It did have a little more memory, but my server was running a full WordPress site as well as some CGI scripts. These days websites tend to include more dynamic content and larger files, but if you are using it for a personal website then it may be sufficient.

Debian Linux

This is based on the Debian Raspberry Pi image from Raspberry Pi download page.

To follow this then the Raspberry Pi will need to have an Internet connection. These instructions assume that it is physically connected to a home router.

Security

The first priority is to make the Raspberry Pi a little more secure. The image includes a default username and password, which once connected to the Internet would allow anyone to login and have free roam of the device.

To change the password for the pi user after logging in issue

```
passwd
```

and follow the prompts for changing the password.

You may also want to add your own username. I have used user1 as the username, but typically this will be a persons name. You can skip this and go straight to the performance / networking steps if this is not required.

This will add a new user and change their password.

```
sudo useradd -m user1

sudo passwd user1
```

Here you will see the first use of the sudo command which we will be using a lot in this. As used above the sudo command allows the user to issue a command as the superuser. Without using the sudo command this would fail as regular users are not allowed to create other users. This is a security feature that protects the system from other users, but also limits the amount of damage that a user can do by mistake (although if prefixed with the sudo command it will not help against accidents).

The new user will need to be added to certain groups to allow the same privileges that the pi user had.
You can add the new user to the groups using the usermod command or you can edit the file directly. I've done the following by editing the file so that you can see the file (it's also arguably a little quicker as you can make multiple changes whilst editing the file). Please be aware that when editing files like these a mistake can result in not being able to login.

There are two command line text editors. The nano editor is the easiest for new users (so that's what I've referred to below), but I do recommend learning the vi text editor as it is useful tool that is installed on all linux systems. If you are familiar with vi then replace nano with vi for the rest of this guide.

```
sudo nano /etc/group
```

Go through the file adding ,user1 to the end of all of the groups that pi is in.
eg

```
adm:x:4:pi,user1
```

Use CTRL-O to save and CTRL-X to quit after editing the file.

The most important is the admin entry as without that the user will not be able to run sudo and hence perform any system administration. Of course if you want to add a different user and don't want to give them admin access then you don't need to make any updates to the /etc/group file.

type

```
exit
```

to logout and now login under the new username to check that it is working correctly.

By default the shell for the new user is the bourne shell. The bash shell is an improvement on that allowing the user of the arrow keys on the command line and autocompletion.

To set the default shell for you new account (when logged in under that account) use:

```
chsh -s /bin/bash
```

You could now remove the pi username if it is no longer required.

```
userdel pi
```

~~Although at the time of writing the current image had an error in the passwd file – which will need to be fixed using the pwck command first.~~

This is just the initial stages in making the Pi more secure. There are other aspects to Linux security including making sure that appropriate security fixes are applied as they become available (eg. apt-get update).

Performance tuning the operating system

Performance tuning is something that you would normally leave until later, but in the case of the Raspberry Pi there is an single option that can be done to improve performance for servers. By configuring it here we can let it get picked up by the reboot later saving us from having to reboot the server.

The Raspberry Pi has 256Mb (or 512Mb for later versions) of RAM. This RAM is however shared between the graphics and main system memory. By default 64Mb is allocated to graphics. This is overkill if you don’t plan to run the graphical interface (or rarely) as in the case of a server. To reduce the amount of memory available for graphics to 32MB enter the following command.

`sudo cp /boot/arm224_start.elf /boot/start.elf`, or use `sudo raspi-config` to do this using the config menu

(you can restore the previous configuration sudo cp /boot/arm192_start.elf /boot/start.elf)

This requires a reboot to take effect, but we will be rebooting later, so there is no need to reboot at this point.

Setting up networking

The next step is to give the Raspberry Pi an static IP address. This is a little more complex as it depends upon your own setup and what router you have on how to achieve this.

By default the Raspberry Pi will request a dynamic IP address which is issued by your router as required. This however may change in future which would make it hard to connect to the webserver. Instead we provide it with an address that doesn’t change such as 192.168.1.4.
Note that this address can be used on the local network, but not on the Internet – later we’ll look at providing access through your router / firewall from the Internet.

First find out what DHCP address has been allocated by using the ifconfig command – see the extract below

```
...
eth0      Link encap:Ethernet  HWaddr b8:27:eb:8a:71:a0
          inet addr:192.168.1.110  Bcast:192.168.1.255  Mask:255.255.255.0
```

http://www.penguintutor.com/linux/raspberrypi-webserver

```
...
```

This is saying that the ethernet port 0 – has an IP address of 192.168.1.110

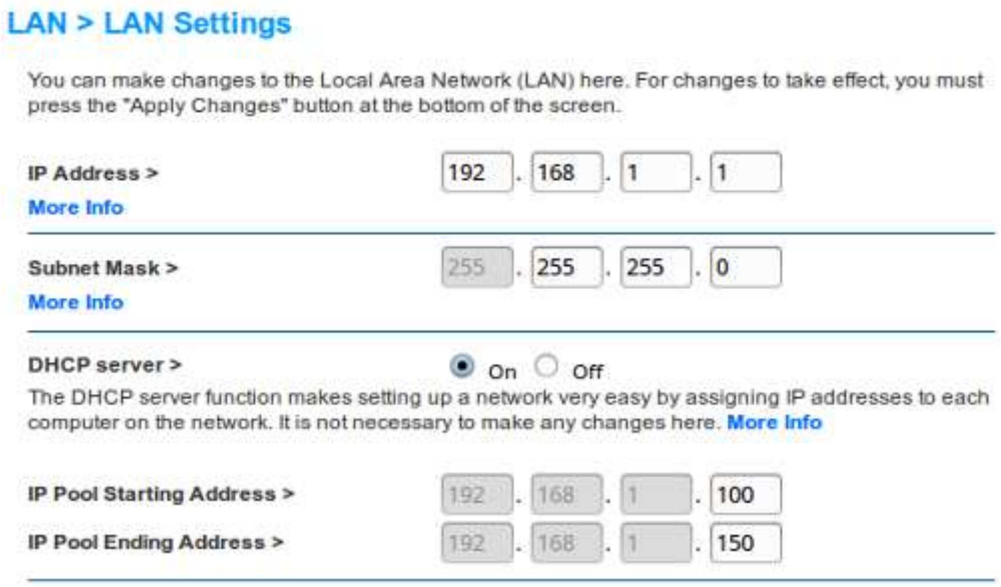
You will also need to find out what address your router is, using the route command

```
$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.1.1     0.0.0.0         UG    0      0        0 eth0
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
```

This shows that the router IP address (Gateway) is 192.168.1.1 and all traffic is sent via that router.

At this point you will also need to check on what address range is being issued by the router. This depends upon the individual router. In my case I have a Belkin Wireless Router which can be reached by pointing a web browser to the IP address of the router 192.168.1.1

The LAN settings are shown below:



LAN > LAN Settings

You can make changes to the Local Area Network (LAN) here. For changes to take effect, you must press the "Apply Changes" button at the bottom of the screen.

IP Address > More Info	192 . 168 . 1 . 1
Subnet Mask > More Info	255 . 255 . 255 . 0
DHCP server >	[✓ On] Off
IP Pool Starting Address >	192 . 168 . 1 . 100
IP Pool Ending Address >	192 . 168 . 1 . 150

The DHCP server function makes setting up a network very easy by assigning IP addresses to each computer on the network. It is not necessary to make any changes here. More Info

In this case the local network has valid addresses from 192.168.1.1 to 192.168.1.254. The router is at address 192.168.1.1 and any DHCP requests will be given entries between 192.168.1.100 and 192.168.1.150 (you can change the range of the DHCP addresses if required). I have used 192.168.1.4 for this server.

To change to static IP address

```
cd /etc/network
sudo nano interfaces
```

replace the line “iface eth0 inet dhcp” with

```
iface eth0 inet static

address 192.168.1.4

netmask 255.255.255.0

gateway 192.168.1.1
```

You should also take a look at the file /etc/resolv.conf

and check it has a nameserver entry (probably pointing at your default gateway)

```
nameserver 192.168.1.1
```

Alternatively you could point directly at your ISPs DNS servers rather.

Whilst you can dynamically reload the network interface I suggest a reboot at this stage to make sure that the configuration is correct.

```
sudo reboot
```

After logging in check using ifconfig to confirm that we have a static ip address

```
...

eth0      Link encap:Ethernet  HWaddr b8:27:eb:8a:71:a0

          inet addr:192.168.1.4  Bcast:192.168.1.255  Mask:255.255.255.0

...
```

Enabling ssh

SSH (Secure Shell) is a network protocol that allows you to login and control the computer through the command line remotely. As the name suggests it is secure as it encrypts communication across the network (so that others cannot see your password etc). It also has some great features such as tunnelling, which we won’t go into here.

The ssh server is installed on the default image and is started by default. You can enable/disable the ssh server using `sudo raspi-config`.

You can now connect to the Raspberry pi remotely (on the same network) via ssh.
If you have another linux computer on the network then from a terminal run

```
ssh 192.168.1.4
```

which will login with the same username. If you want to use a different username then prefix that before the ip address with an @ sign.
eg

```
ssh user1@192.168.1.4
```

I believe that ssh is also installed on a MAC so you can use the same commands as above.

If you want to connect from Windows then there are several options, but I suggest the open source software Putty.

Making the server available on the Internet

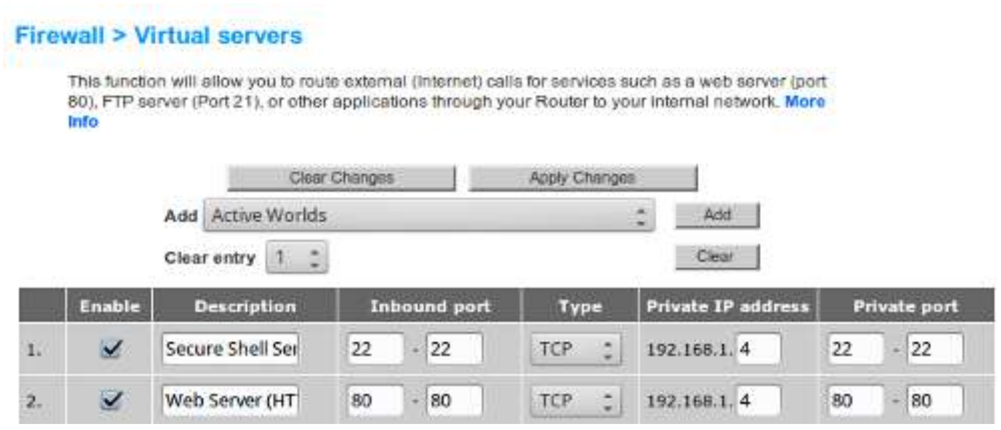
Next we are going to configure the router to allow ssh logins and web traffics through its firewall to the Raspberry Pi.

You did remember to change the default password for the pi username didn't you! If you haven't already changed the default password then do it now otherwise anyone will be able to login to your Raspberry Pi.

As a home user the ip address used on your local network is a private address range that will not work over the Internet. Instead your ISP will provide a single dynamic IP address which is used by the router. To allow traffic to flow from the Internet to your Raspberry Pi needs the IP address of the Pi to be made to look as though it is from the router. This is a process called Network Address Translation (NAT).

The ports that need to be allowed through are port 80 (http) and if you would like to be able to login to the computer from the Internet then port 22 (ssh).

To do this you will need to consult the instructions on your router. In the case of my Belkin router this is through the Firewall > Virtual servers settings (see below), but Netgear this is Advanced > Security > IP Forwarding.



Firewall > Virtual servers

This function will allow you to route external (internet) calls for services such as a web server (port 80), FTP server (Port 21), or other applications through your Router to your internal network. More Info

Add: Active Worlds

Clear entry: 1

	Enable	Description	Inbound port	Type	Private IP address	Private port
1.	✓	Secure Shell Ser	22 - 22	TCP	192.168.1.4	22 - 22
2.	✓	Web Server (HT	80 - 80	TCP	192.168.1.4	80 - 80

DNS

The final stage is to have a DNS entry point at your router's IP address. In my case I have cable Internet through Virgin Media. Although it does have a dynamic IP address the address does not normally change. I have a static DNS entry on a Internet DNS server. The entry only needs to be changed about once every year or when Virgin Media perform significance maintenance on the Internet connection.

If you have a dynamic IP address that changes on a more recent basis then you will need to register for a dynamic dns service.

http://www.penguintutor.com/linux/raspberrypi-webserver

Install apache webserver

The Apache webserver is available to download from the Debian repositories. This can be done through the apt tools.

First have you refreshed the software repositories? If not run `sudo apt-get update` to make sure that it knows about any new packages / versions available.

Apache is installed by entering the following

```
sudo apt-get install apache2
```

Install mysql

The mysql database server is also available through the Debian repositories and installed as

```
sudo apt-get install mysql-server
```

During the install there is a prompt request for a password.
The password is for the mysql root user.

Install PHP

Perl is installed as part of the operating system so I will just be adding PHP.

The following commands will install PHP version 5 and the mysql libraries to allow PHP to access the mysql database.

```
sudo apt-get install php5

sudo apt-get install php5-mysql
```

Setup complete

Once the setup is complete you can access the web page by pointing your browser to the router IP address or DNS entry.

You should get a page back stating that it works, but that there is no content loaded.

To test that the webserver and PHP are working correctly then delete the file /var/www/index.html and create a file /var/www/index.php with the contents of this page.

Note that the filename of the link ends in .txt to prevent my webserver from running this, you should have the file ending with .php so that the file is run as a PHP script.

Running a lightweight webserver on the Raspberry Pi (lighttpd)

This guide covers setting up a light webserver on Linux using the lighttpd web server on the Raspberry Pi. Most of these instructions can also be applied to other Debian or Ubuntu based distributions (except the tasks using the raspi-config tool). I've already written a tutorial based on the "full-fat" Apache webserver (running a LAMP Apache based webserver on the Raspbbery Pi).

Lighttpd provides a way of setting up a web server without putting too much load on the limited processing capability. It's ideal for providing web access to the Raspberry Pi as a monitoring tool, or as a lightweight webserver for a personal website.

Debian Linux

This is based on the Debian Raspberry Pi Raspbian.

One of the advantages of Debian Wheezy on the Raspberry Pi is that ssh is enabled by default and that the raspi-config tool provides a convenient way to increase the size of the root partition (useful if you have an SD card bigger than 2GB) and to change the default password for the pi user (essential for everyone). The raspi-config tool should run when you first start the system, otherwise use sudo raspi-config.

Using the command line and editing files

As we are going to be doing this through the command line it is useful to understand a little about the shell. Although this provides step-by-step instructions if you haven’t used the command line previously I suggest you read the basic shell reference guide, and then return to this point.

Throughout the install you will see many commands prefixed with sudo. The sudo command allows the user to issue a command as the superuser (or in certain circumstances as another user). Without using the sudo command many of the commands would fail stating insufficient permissions. This is a security feature that protects the system from other users, but also limits the amount of damage that a user can do by mistake (although if prefixed with the sudo command it will not help against accidents).

Static network IP address

The next step is to give the Raspberry Pi an static IP address. This is a little more complex as it depends upon your own setup and what router you have on how to achieve this.

By default the Raspberry Pi will request a dynamic IP address which is issued by your router as required. This however may change in future which would make it

hard to connect to the webserver. Instead we provide it with an address that doesn't change such as 192.168.1.4.
Note that this address can be used on the local network, but not on the Internet – later we'll look at providing access through your router / firewall from the Internet.

First find out what DHCP address has been allocated by using the ifconfig command – see the extract below

```
...
eth0      Link encap:Ethernet  HWaddr b8:27:eb:8a:71:a0
          inet addr:192.168.1.110  Bcast:192.168.1.255  Mask:255.255.255.0
...
```

This is saying that the ethernet port 0 – has an IP address of 192.168.1.110

You will also need to find out what address your router is, using the route command

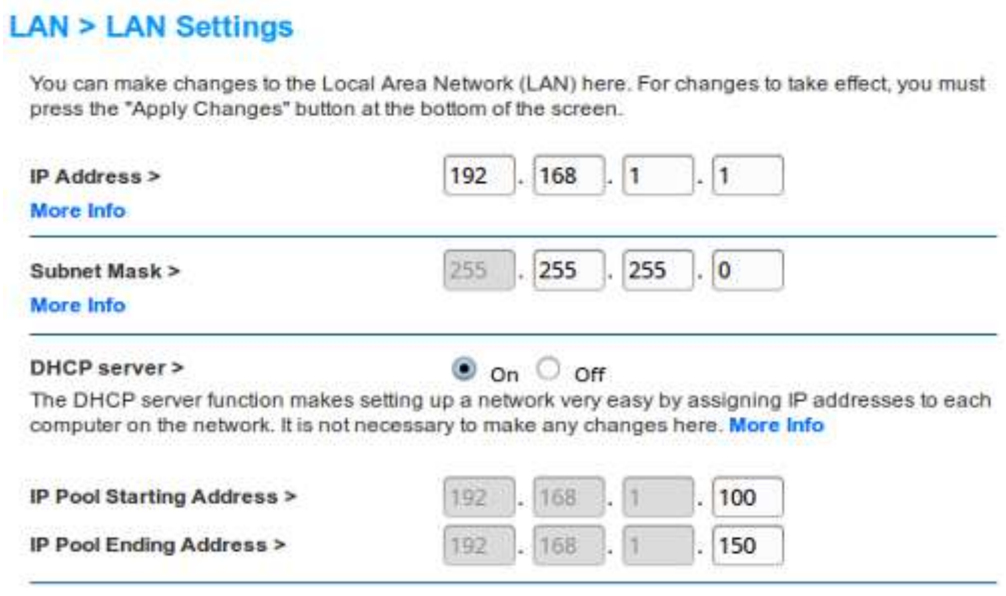
```
$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.1.1     0.0.0.0         UG    0      0        0 eth0
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
```

This shows that the router IP address (Gateway) is 192.168.1.1 and all traffic is sent via that router.

At this point you will also need to check on what address range is being issued by the router. This depends upon the individual router. In my case I have a Belkin Wireless Router which can be reached by pointing a web browser to the IP address of the router 192.168.1.1

The LAN settings are shown below:

http://www.penguintutor.com/linux/raspberrypi-webserver


LAN > LAN Settings

You can make changes to the Local Area Network (LAN) here. For changes to take effect, you must press the "Apply Changes" button at the bottom of the screen.

IP Address >	192 . 168 . 1 . 1
Subnet Mask >	255 . 255 . 255 . 0
DHCP server >	[✓ On] Off
IP Pool Starting Address >	192 . 168 . 1 . 100
IP Pool Ending Address >	192 . 168 . 1 . 150

The DHCP server function makes setting up a network very easy by assigning IP addresses to each computer on the network. It is not necessary to make any changes here.

In this case the local network has valid addresses from 192.168.1.1 to 192.168.1.254. The router is at address 192.168.1.1 and any DHCP requests will be given entries between 192.168.1.100 and 192.168.1.150 (you can change the range of the DHCP addresses if required). I have used 192.168.1.4 for this server.

To change to static IP address

```
cd /etc/network

sudo nano interfaces
```

replace the line "iface eth0 inet dhcp" with

```
iface eth0 inet static

address 192.168.1.4

netmask 255.255.255.0

gateway 192.168.1.1
```

You should also take a look at the file /etc/resolv.conf

and check it has a nameserver entry (probably pointing at your default gateway)

```
nameserver 192.168.1.1
```

Alternatively you could point directly at your ISPs DNS servers rather.

Whilst you can dynamically reload the network interface I suggest a reboot at this stage to make sure that the configuration is correct.

```
sudo reboot
```

After logging in check using ifconfig to confirm that we have a static ip address

```
...
```

```
eth0      Link encap:Ethernet  HWaddr b8:27:eb:8a:71:a0

          inet addr:192.168.1.4  Bcast:192.168.1.255  Mask:255.255.255.0
...
```

Using ssh

SSH (Secure Shell) is a network protocol that allows you to login and control the computer through the command line remotely. As the name suggests it is secure as it encrypts communication across the network (so that others cannot see your password etc). It also has some great features such as tunnelling, which we won't go into here.

Using the Wheezy image ssh is turned on by default. This can be enabled / disabled through the raspi-config tool if required.

You can now connect to the Raspberry pi remotely (on the same network) via ssh.
If you have another linux computer on the network then from a terminal run

```
ssh 192.168.1.4
```
which will login with the same username. If you want to use a different username then prefix that before the ip address with an @ sign.
eg
```
ssh pi@192.168.1.4
```

If you want to connect from Windows then there are several options, but I suggest the open source software Putty.

Making the server available on the Internet

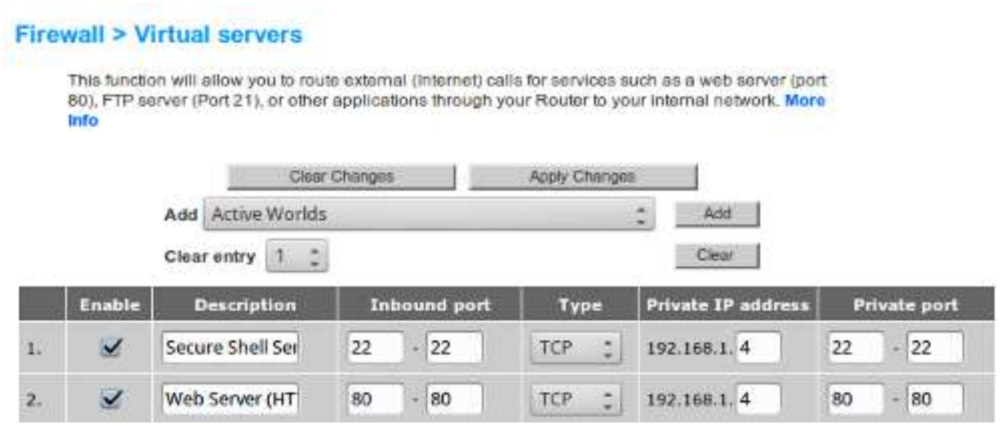
Next we are going to configure the router to allow ssh logins and web traffics through its firewall to the Raspberry Pi.

You did remember to change the default password for the pi username didn't you! If you haven't already changed the default password then do it now otherwise anyone will be able to login to your Raspberry Pi.

As a home user the ip address used on your local network is a private address range that will not work over the Internet. Instead your ISP will provide a single dynamic IP address which is used by the router. To allow traffic to flow from the Internet to your Raspberry Pi needs the IP address of the Pi to be made to look as though it is from the router. This is a process called Network Address Translation (NAT).

The ports that need to be allowed through are port 80 (http) and if you would like to be able to login to the computer from the Internet then port 22 (ssh).

To do this you will need to consult the instructions on your router. In the case of my Belkin router this is through the Firewall > Virtual servers settings (see below), but Netgear this is Advanced > Security > IP Forwarding.



Firewall > Virtual servers

This function will allow you to route external (Internet) calls for services such as a web server (port 80), FTP server (Port 21), or other applications through your Router to your internal network. More Info

	Enable	Description	Inbound port	Type	Private IP address	Private port
1.	✓	Secure Shell Ser	22 - 22	TCP	192.168.1. 4	22 - 22
2.	✓	Web Server (HT	80 - 80	TCP	192.168.1. 4	80 - 80

DNS

The final stage is to have a DNS entry point at your router’s IP address. In my case I have cable Internet through Virgin Media. Although it does have a dynamic IP address the address does not normally change. I have a static DNS entry on a Internet DNS server. The entry only needs to be changed about once every year or when Virgin Media perform significance maintenance on the Internet connection.

If you have a dynamic IP address that changes on a more recent basis then you will need to register for a dynamic dns service.

Installing lighttpd

To install the lighttpd web server issue the command.

```
sudo apt-get install lighttpd
```

This will install the web server and also pull in any other packages (called dependencies) that are required. The server will be automatically started and set to start by default after a reboot.

```
[ ok ] Starting web server: lighttpd.
```

Install mysql database (optional)

Whilst you can have a perfectly good website without a database, database provide a good way of holding data and are a requirement for many content management systems (CMS) and web based applications. If you don't need a databse the you can skip this and go straight to configuring php.

Mysql is the most popular database server, whilst there are other alternatives some of which may require less resources most third party software for Linux is designed to use Mysql.If required it can be installed using

```
sudo apt-get install mysql-server
```

During the install there is a prompt request for a password.
The password is for the mysql root user.

Install PHP

Perl is installed as part of the operating system so I will just be adding PHP, which is an interpretted programming language. Whilst this is optional it is recommended that it is installed as it is particularly useful.

The following commands will install PHP version 5.

```
sudo apt-get install php5-common php5-cgi php5
```

Note it's important to install in the order listed above. If you try to install php5 without first installing the php5-cgi package then it will install Apache as well, which we don't want for this light-weight lighttpd server.

If you installed mysql then you should also issue the following command to install the php mysql libraries to allow PHP to access the mysql database.

```
sudo apt-get install php5-mysql
```

To enable the server to handle php scripts the fastcgi-php module should be enabled by issuing in the command

```
sudo lighty-enable-mod fastcgi-php
```

Then reload the server using

```
sudo service lighttpd force-reload
```

Set permissions on the web directory /var/www/

It is useful to change the permissions on the www directory to allow your user to update the webpages without needing to be root.

Change the directory owner and group

```
sudo chown www-data:www-data /var/www
```

allow the group to write to the directory

```
sudo chmod 775 /var/www
```

Add the pi user to the www-data group

```
sudo usermod -a -G www-data pi
```

You should logout and back in - to pick up group permissions, or if running X you can just start a new terminal.

Testing the server

Once the setup is complete you can access the web page by pointing your browser to the router IP address or DNS entry.

You should get a page back stating that it works, but that there is no content loaded.

To test that the webserver and PHP are working correctly then delete the file /var/www/index.lighttpd.html and create a file /var/www/index.php with the contents of this page.

http://www.penguintutor.com/linux/raspberrypi-webserver

Note that the filename of the link ends in .txt to prevent my webserver from running this, you should have the file ending with .php so that the file is run as a PHP script.