Setting up a Web Server using Raspbian Wheezy on the Raspberry Pi
(© Kenneth A Spencer)

To use the Raspbian Wheezy distribution of the Debian Linux release of Wheezy, for the Raspberry Pi, as a web server requires changes to the basic configuration and the installation of additional software. It is also advised that once these processes have been carried out, a full backup copy of the host SD Card should be made for easy restoration, and this topic is also discussed.

This document explains fully how to make your Raspberry Pi ready to function as a Web Server with database and PHP programming support. It is in three parts. Part A deals with obtaining, installing and configuring the software which you will need, and ensuring that it is all working. Part B discusses backup methods. Part C explains and illustrates some ways of making valuable use out of the machine and the software which has been installed.

Part A: Obtaining, Installing and Configuring the Software

A1. Install RaspbianWheezy
The 20121018 distribution of RaspbianWheezy can be installed using `Win32DiskImager.exe` having copied an image of the distribution image into the same location as the executable.
New images are available from time to time from *www.raspberrypi.org/downloads*.

A2. Boot the Image
This distribution is the first to have the `Raspbi-Config` program, which starts on first boot. Ensure that the keyboard type, keyboard layout, keyboard language, locale and time zone are correctly selected. Then ensure that `ssh` is configured for startup, so as to allow remote login when required, and that the option is selected for expansion of the root filing system to occupy the full capacity of the SD Card.
The configuration program also offers a range of over-clocking options from a conservative increase of CPU frequency by 100MHz to a maximum of 300MHz combined with CPU voltage and RAM frequency increases.
Note that the root filing system will be expanded at the next boot, and that this process will take an hour at least, although it will vary with your SD Card size. There is no indication of progress, so just wait!

A3. Set a root password
Once the machine is rebooted after stage A2, execute the command:
```
      sudo passwd
```
and enter your preferred root password. As the installations and reconfigurations are best carried out with root privilege you should then issue the command:
```
      sudo su
```
which is the easiest way to change to the privilege level required for the rest of your session.

A4. Install the `proftpd` ftp server
Execute the following command to obtain and install the `proftpd` ftp sever:
```
      apt-get install proftpd
```
You will use `proftpd` later on to transfer your website files to the newly configured web server.
Once `proftpd` is installed, it is easier to continue with the configuration by remote login at a PC or Mac workstation. Use an `ssh` terminal client such as `PuTTy` which is readily available free-of-charge. Download and install it onto your PC. Then run it and enter the IP address of your Raspberry Pi so as to login. To get the IP address, execute the command `ifconfig` directly at the Raspberry Pi (see A5.1).
Once logged in remotely with `PuTTy`, you will be able to update the RaspbianWheezy installed components, install the `Apache2` web server, install the `PHP5` programming environment, install the `MySQL` database, install `APC` support for `PHP`, and finally install the `nmap` utilities.
Collectively these components constitute the so-called LAMP server and their installation is described in the succeeding sections 5 to 9 below.

A5. Install the `Apache2` web server

If you wish, you can copy and paste the commands from the following paragraphs by highlighting the text of each command, tapping [Ctrl]+C (to copy the highlighted text), and then right-clicking over the cursor in the `PuTTy` window.

Before installing any more software packages, update your system fully by executing the following:

```
      sudo apt-get update && sudo apt-get upgrade
```

Then execute the following command to obtain and install the `Apache2` web server:

```
      apt-get install apache2
```

Apache2 will be running after installation, but if you need to start or stop it, the commands are:

```
      service apache2 start
      service apache2 stop
```

A5.1 After the initial installation, `Apache2` may report the following warning when started or stopped:
Could not determine the server's fully qualified domain name, using 127.0.X.1 for ServerName
This means that `Apache2` is not quite correctly configured for your individual Raspberry Pi.
To resolve this, first identify and note down the IP address of your Raspberry Pi machine (also explained in 4. above):

```
      ifconfig
```

Record the numbers immediately following `inet addr:` in the first paragraph of the output.
Then, edit two existing files, and create an new file as detailed in A5.1a below.

A5.1a Edit `/etc/hostname` as follows:

```
      cd /etc
      nano hostname
```

`nano` is a Linux editor. Change the line `raspberrypi` to your chosen name for your server.
Mine is `raspbianwheezy97`. This is essential anyway if you have more than one Raspberry Pi.
Then save the file with `[Ctrl]+O` and exit `nano` with `[Ctrl]+X`.

A5.1b Then edit `/etc/hosts` as follows:

```
      nano hosts
```

Add the lines (the first line may already present but commented out with a # symbol):

```
      127.0.0.1 localhost
      10.0.0.97 localhost
```

and, if you wish, also add the following line to the `hosts` file:

```
      10.0.0.97 raspbianwheezy97
```

In the `/etc/hosts` file, `127.0.0.1` is the standard IP address and `localhost` the standard TC/IP hostname for local reference to the current physical machine.
Replace `10.0.0.97` by your actual IP address noted earlier, and `raspbianwheezy97` by your chosen name for your Raspberry Pi machine.
Then save the file with `[Ctrl]+O` and exit `nano` with `[Ctrl]+X`.

A5.1c Next, create an extra file in the `/etc/apache2/conf.d` directory as follows:

```
      cd /etc/apache2/conf.d
      nano servername.conf
```

which will open the `nano` editor with a new empty file called `servername.conf`.
Type your `ServerName` into the file under the `ServerName` directive:

```
      ServerName     raspbianwheezy97
```

Change the name to that chosen for your own machine.
Then save the file `[Ctrl]+O` and exit `nano` with `[Ctrl]+X`.
As `Apache2` loads data from the files in `conf.d` into its configuration, these changes will remove the error concerning `Apache2` not finding your server name. It is preferable to use this method of adding to the `Apache2` configuration as editing the main configuration file may introduce unintentional errors.
Stop and start `Apache2` as described in A5. above.

A5.2 After restarting `Apache2` you can test `Apache2` from your remote workstation by typing *http://10.0.0.97* into the address bar of a web browser (e.g. Internet Explorer). Replace *10.0.0.97* with your own Raspberry Pi's IP address.
You should see a message stating "*It works!*"

If you wish to be able to type *http://raspbianwheezy97* into the address bar, then you must add

```
	10.0.0.97 raspbianwheezy97
```

into your Windows `hosts` file, which is commonly in `C:\Windows\System32\drivers\etc.`
By now you know that you must use your own Raspberry Pi IP address and chosen hostname.
If you have a Domain Name Server (DNS) on your network, it is preferable to record the Raspberry Pi's details in that rather than in the `hosts` files of all of your workstations.

A6. Install the PHP programming environment

`PHP` is a very useful server-side programming or scripting language which amongst many other things allows your web pages to use the `MySQL` database management system to load data into your pages, and to save data from your pages into your database.

Execute the following command to obtain and install the `PHP` programming environment version 5:

```
	apt-get install php5 libapache2-mod-php5 php5-intl php5-mcrypt
php5-curl php5-gd php5-sqlite
```
[NB: all on a single line]

If you wish, you can create a simple web page to show the PHP configuration information:

```
	cd /var/www
```

to get to the directory containing your web page. Then open `nano` with an empty file:

```
	nano phpinformation.php
```

When it opens, type in the following:

```
<?php
phpinfo();
?>
```

Then save the file with `[Ctrl]+O` and exit `nano` with `[Ctrl]+X`.
View your new page by typing *http://raspbianwheezy97/phpinformation.php* into your browser address bar, remembering to substitute your Raspberry Pi's IP address or host/server name for mine.

A7. Install the MySQL database management system

Execute the following command to obtain and install the `MySQL` database management system:

```
	apt-get install mysql-server mysql-client php5-mysql
```

During the installation, you will be invited to enter a password for the top level `MySQL` user. The username is commonly `root`, but the password is up to you. Note that this is the managing user for `MySQL` and is not related to the `root` user of your Raspbian Wheezy operating system.

Then to take account of system and configuration changes, stop and start the `Apache2` web server.

You can test your `MySQL` installation by logging in to `MySQL`. You will have to use the `MySQL` `root` user for the moment:

```
	mysql -u root -p
```

The `-u` parameter signifies that the next item is a username, and the `-p` parameter tells `MySQL` to expect a password. When prompted, enter the `root` password created earlier, and `MySQL` will respond with an Oracle copyright notice and the `mysql>` prompt. Exit `MySQL` by typing `exit`.

It is very much easier to manage a database, including the creation of tables and other database objects in `MySQL` using `phpMyAdmin`, and we will install that most useful tool very shortly. But we have two or three more steps to complete first.

A8. Install the APC support package for PHP

`ACP` is an alternative caching support system for `PHP`, which enables `PHP` intermediate code to be cached. This can improve the performance of `Apache2` and other programs which may execute `PHP` code.
Execute the following command to obtain and install the APC support system for PHP:

```
      apt-get install php-pear php5-dev apache2-prefork-dev build-
essential make && pecl install apc
```
[NB: all on a single line]

Then edit the php.ini configuration file:

```
      nano /etc/php5/apache2/php.ini
```
and add the following text to the file, in the `Dynamic Extension` section, some way down the file:

```
      extension=apc.so
```
Then to take account of system and configuration changes, stop and start the `Apache2` web server.

A9. Install some additional items

There are a few additional items which may be usefully installed before we install the next major item.
Execute the following command to obtain and install them:

```
      apt-get install nmap
```
It installs some additional fonts, the `nmap` network analysis tool some other utilities. The `nmap` program has a GUI interface, (on your Pi only) called `zenmap` which can be installed by:

```
      apt-get install zenmap.
```

A10. Install `phpMyAdmin`

`phpMyAdmin` is one of the most useful tools for working with the `MySQL` database management system. It is a web-based control panel, but it is not only that. It also includes support for creating and amending your database tables, for entering data into them, and for designing and executing queries.

Execute the following command to obtain and install `phpMyAdmin`:

```
      apt-get install libapache2-mod-auth-mysql php5-mysql phpmyadmin
```
When asked if `phpMyAdmin` should configure a `MySQL` database for itself, select `yes`.
When asked, enter the password of the `root` `MySQL` user, so as to authenticate yourself on `MySQL`.
Once authenticated on `MySQL` you will be asked to create and confirm a password for `phpMyAdmin`.
Then you'll be asked which server on which to install it: select `Apache2`.
During the installation, you will be offered several options: accept the default suggestions at this stage of your knowledge! After further processing, the `phpMyAdmin` installation will finish.

There are two adjustments to the configurations necessary before you can use `phpMyAdmin`.
First edit the `php.ini` file to include a `MySQL` library:

```
      nano /etc/php5/apache2/php.ini
```
Then type the following text into the `Dynamic Extensions` section about two thirds down the page:

```
      Shellextension=mysql.so
```

As things stand, there is no provision for showing the `phpMyAdmin` pages from your default web page directory. To such an create an entry make a symbolic link in the `Apache2` data directory, as follows:

```
      cd /var/www
      ln -s /usr/share/phpmyadmin
```

Now from a workstation, you'll be able to navigate to:

http://raspbianwheezy97/phpmyadmin

or directly on your Pi to:

http://localhost/phpmyadmin

If logging in from a workstation, remember to substitute your own IP address or if you have entered it into your DNS server or the workstation's `hosts` file, you may enter your `hostname`.

When invited to login on the `phpMyAdmin` opening page, respond with the username `root` and the password which you created during `phpMyAdmin` installation.

A11. Install the `Webmin` administration Tool
`Webmin` is an extremely useful tool for administering all aspects of a Linux server.

Before installing `Webmin`, it is necessary to edit the `sources.list` file:

```
     nano /etc/apt/sources.list
```

Add the following two lines:

```
     deb http://download.webmin.com/download/repository sarge contrib
     deb http://webmin.mirror.somersettechsolutions.co.uk/repository
sarge contrib
```

[NB: the second entry must be all on a single line]

Then you must obtain and install the GPG key with which the repository holding `Webmin` is signed:

```
     cd /root
     wget http://www.webmin.com/jcameron-key.asc
     apt-key add jcameron-key.asc
```

Then get any system updates, following which you can obtain and install the `Webmin` package(s):

```
     apt-get update
     apt-get install webmin
```

It may take some time, but all dependencies should be resolved automatically.

`Webmin` runs it's own secure sockets web server, separate from `Apache2`. It does this using its own port number (10000) which must be added to the `Webmin` address when typed into your web browser address bar: *https://10.0.0.97:10000/.* The *https://* in the address bar indicates that the web browser is asking for the page using secure sockets layer, which is a more secure protocol than the usual *http://*. Of course, you can also use the form *https://yourPiHostname:10000* for the `Webmin` address if you wish and if you have your Pi details in the workstation `hosts` file or in a DNS server..
When invited to login, you will normally respond as user `root` with your `root` password, or as any user who can use `sudo` to run commands as `root`.

A12. Install the Webalizer Graphical Webstats package
By default, the `Apache2` web server generates a line of data in a log file for each request it receives to serve a webpage. This data consists of a series of fields containing such items as the IP address of the client requesting the page, the date and time, the filename of the page, the browser and operating system, and more. You can read this data into a database for analysis, but it can also be converted into a graphical display which is useful for less formal study of the use of your web server activity.

You can obtain and install `Webalizer` by executing the following command:

```
     apt-get install webalizer
```

The default configuration of `Webalizer` is not quite correct for `Apache2` under Raspbian Wheezy. Therefore you must edit the `Webalizer` configuration file `/etc/webalizer/webalizer.conf`

```
     nano /etc/webalizer/webalizer.conf
```

In the `webalizer.conf` file, for the setup of `Apache2` so far, you may need to locate and edit the `Apache2` log file entry to:

```
     logfile   /var/log/apache2/access.log
```

Later, if you decide to implement multiple virtual hosts on `Apache2` this entry will need further amendment.
In addition, it appears that `Webalizer` does not install the default geographical IP locator database, but instead installs `GeoIP`. Therefore edit the `webalizer.conf` file further:
Uncomment/edit the lines:

```
     GeoIP           yes
     GeoIPDatabase   /usr/share/GeoIP/GeoIP.dat
```

Part B: Backup your installation

Now that you have installed and tested the Debian variant of Linux known as Rasbian Wheezy, the ftp server `proftpd`, the `Apache2` web sever, the `PHP` programming environment and its support system `APC`, the `MySQL` database management system, `phpMyAdmin` `MySQL` management tool, and the `Webmin` general server management tool, you would be well-advised to make a backup of the SD Card.

We have tried many methods of backup of the SD Card contents. Most of them appear to work, but in the event, do not actually enable a safe and reliable restoration. For example:

B1. Win32DiskImager.exe and DiskImage_1_6_WinAll

Your Raspberry Pi can be backed up by making an image of the SD Card using one of the programs. You can use the program to extract the SD Card image by plugging your card into your PC or Mac and selecting the option that copies the SD Card image into the same location.

However, we have noted the following problem: the size of the backup image produced sticks at 4GByte even for a 64GByte SD Card. This cannot be right!

You can restore the image to the SD Card using the same procedure as writing the OS originally: select the card from the drop down box and then selected the image file to be restored.
However, we have noted the following problems:
- on many SD Cards, the programs may report that the target SD Card is too small, or other errors;
- after an apparently successful restoration the card does not boot properly, reporting errors and hanging.

There are several similar programs, and they also seem to fail, at least with larger cards.

B2. The Best Backup Solution

We have found that a working backup system is that from **DiskInternals**, named **Linux Reader**. This program will locate Linux discs and partitions on Linux discs and permit copies of whole discs or partitions to be created on any other medium present. Note that for very large SD Cards, you will need lost of space on the selected medium.

Once created, your backup disc image can be restored to an SD Card using a program such as **Active Disk Image**, or **WinImage Administrator**. We have successfully backed up a 64GByte Raspberry Pi Linux SD Card image and restored it to a 128GByte SD Card using this method. We have not tried restoration to a similar size SD Card as the original source card.

Please note that whilst backups are vital to save time and preserve data, it is unwise to test the backup on your working SD Card, as it is not unusual for such backup or the restoration, to fail or to be irretrievable. It is also essential to test the backup - restore sequence on a card with as much data on it as you can usefully get.

Part C: Making Serious Use of the Installed Software

Having spent the last several hours installing a powerful suite of server programs onto your Raspberry Pi, the next task is to start to use the machine seriously.

C1. Copy your Website Files

Actually developing a website is out of the scope of this article as it is not strictly related to the setting up of your Raspberry Pi. Therefore we are making the assumption that you already have a website, complete with its content, and you would like to host it on your Raspberry Pi using the software infrastructure which we have so far installed and setup.

So, if you have already developed website, you can copy it onto your Raspberry Pi. To do this, follow this procedure:
Use an FTP Client such as `WS_FTP95` or similar, to connect to your Raspberry Pi - a graphical FTP client is generally much easier for you to use than continuing on the command line.
You will need to enter the IP address of your Pi into your FTP client. If you log into `ProFTPd` with the Pi username and password you may not be allowed to make directories or copy the files. Therefore you will have to extend the default settings of the `ProFTPd` server on your Raspberry Pi to allow `root` login.
NB: if you have extended the `root` login to `ProFTPd` for a session, it would be wise to revert afterwards to the standard settings, which do not permit root to login via FTP remotely.
To permit `root` login to `ProFTPd` open `PuTTy` and login to your Pi as `root`. Then open the `ProFTPd` configuration file in the `nano` editor:

```
      nano           /etc/proftpd/proftpd.conf
```

For tidiness, first of all set the `ServerName` directive to your own host name for your Pi

```
      ServerName     "yourPiHostname"
```

Then locate the `RootLogin` directive and change it to "`On`", or create the line at the end of the file:

```
      RootLogin      On
```

Then save the file `[Ctrl]+O` and exit `nano` with `[Ctrl]+X`.
Finally, you need to make a change to the `ftpusers` file, which contains a list of usernames prohibited from logging in via an FTP client:

```
            nano           /etc/ftpusers
```

The `root` user is commonly at the top of the file. Simply insert a hash symbol (#) at the very beginning of that line to remove the root user from the prohibited list.
Then save the file `[Ctrl]+O` and exit `nano` with `[Ctrl]+X`.
You will need to stop and restart the ProFTPd service in order for the changes to take effect:

```
      service   proftpd   stop
      service   proftpd   start
```

Login to your FTP client program and navigate to your `Apache2` data directory, usually `/var/www`. You must then create a directory for your website - we will name our site `oursite1`. Then navigate to the location where you hold your website files and content and start the process of transfer of all of your website files to your chosen site directory on your Raspberry Pi `/var/www/oursite1`.

You will now be able to view your website from your PC by starting a web browser and typing the following into the address bar:

http://10.0.0.97/oursite1
or *http://raspbianwheezy97/oursite1*

but remember to substitute your Pi IP address for mine, and your Pi hostname for mine, and to enter your Pi hostname into your Windows host file.
Your site is now available, but there are two further enhancements we can make to your setup: use a virtual hostname and make to site available outside your network - i.e. go global!

C2. Create a Virtual Host

The snag with the current setup is that your website is seen as a subdirectory of the website of the hostname. In the case illustrated, `oursite1` is a subdirectory of the main site `raspbianwheezy97`. What you'd prefer to be able to do is to type *http://oursite1* into the address bar of your web browser and have it displayed as though `oursite1` was the actual website URL. That can be done by setting up `oursite1` as a virtual host in your `Apache2` configuration. A virtual host is simply a name assigned as a website URL and hosted on your `Apache2` server, but with a name independent of your machine hostname. You can have more than one virtual host if you wish, but we will set up just one.

First, let's stop the `Apache2` service - you know how to do this by now.
Next, we need to navigate to the location from where `Apache2` will read a new configuration file containing our setup for a virtual host.

```
     cd   /etc/apache2/sites-available
```

You can place all virtual host configuration files here. But each file will not be enabled unless a symbolic link to the file is created in the parallel directory `sites-enabled`. There is a tool for doing this, which we will use once we have created the file.
We will call our configuration file `oursites.conf`, and we will create it with the `nano` editor:

```
     nano oursites.conf
```

Note that the oursites is plural, as we may have more than one virtual site! Then edit the file so that it contains the following text:

```
# oursites.conf
# KA Spencer 201211
#
# First, the listening port (if not specified elsewhere):
Listen    80
#
# Next, the IP address and port for the virtual host. This assumes
#  that you have only one IP address and port for this server.
# Be sure to substitute your own parameters throughout this file!
NameVirtualHost     10.0.0.97:80
#
# Next, add the default server, because creating a virtual host
#  causes Apache2 to ignore the default server configured in the
#  /etc/apache2/sites-available/default file.
# If you do not do this, any html files in /var/www will be ignored!
<VirtualHost   10.0.0.97:80>
DocumentRoot   /var/www
DirectoryIndex index.htm index.html index.php
</VirtualHost>
#
# Next your first virtual server details:
<VirtualHost   10.0.0.97:80>
ServerAdmin    email@youraddress.com
ServerName     oursite1
DocumentRoot   /var/www/oursite1
DirectoryIndex index.htm index.html index.php
ErrorLog       /var/www/oursite1/log/error.log
CustomLog      /var/www/oursite1/log/access.log   combined
</VirtualHost>
# This will allow you to access your "oursite1" website by that name.
```

For the `CustomLog` directive, choose a file format defined in `/etc/apache2/apache2.conf`.

There are one or two other configuration adjustments to make before this will work.
First, create a directory for the `Apache2` log files, as given in the `oursites.conf`:

```
      mkdir     /var/www/oursite1/log
```

Then we must make a link to our new virtual host file in `sites-enabled`. Do this using the `a2ensite` command:

```
      a2ensite  oursites.conf
```

This program creates the link for you. Its sister program `a2dissite` will remove the link if you wish to take your virtual host site offline.

Finally, it is necessary to enter your new virtual host site name into your Raspberry Pi `/etc/hosts` file (see A5.1b), your PC workstation `hosts` file (see A5.2), or into your domain name server (DNS) if you have one.

Now if all is well, you can restart the `Apache2` server in the usual way. There should be no errors on startup. If errors occur, stop the `Apache2` server and examine each stage in turn for errors.
If everything is correct, when you type *http://oursite1* into your web browser address bar, you should see your website.
You have a virtual host working! Wouldn't it be good if the whole world could see you site - that's possible, let's see how.

C3. Going Global

Whether or not you can set up your site to be visible in the Internet at large depends on a number of factors:
- do you have ADSL or Fibre Broadband? If you have only diallup, I must tell you that what we are trying to do is not very reliable under diallup Internet access!
- does your ISP give you a permanent (fixed) IP address? If you do not know the answer to this question, then ask them! Even if you have a dynamic IP address it may still be possible to make your site available, but it is easier with a static address;
- does your router support "Port Forwarding"? Most do;
- does your router support "Hairpin Routing"? Possibly the only way to find out is to try it when your setup is completed. Testing the set up will have to be done on a friend's Internet connection if your router does not support this mode. Absence of Hairpin Routing does not mean that you cannot make your Raspberry Pi website available on the Internet at large, it simply means that it will be out of bounds to your Local Area Network (LAN) as served by your router. Nearly all Belkin ADSL routers, many Netgear cable routers are fine. Some BT ADSL routers do not allow Hairpin Routing.

C3.1 Router Setup

First, let's setup your router for Port Forwarding. This process allows your router to pick up TCP data packets that it receives from the Internet, examine the port number with which the packet is labelled, and then direct the packet to the correct computer on your network using the target computer's IP address. Thus all port 80 packets (the standard HTTP port) can be directed uniquely to your Raspberry Pi. This facility is known variously as Port Forwarding, Virtual Server, Port Mapping or possibly just as Network Address Translation (which is slightly incorrect).

First, you need to login to your router using the username and password that has previously been set up for it. To do this you need the router's LAN IP address. You can find that by opening a Command Window on your PC, and typing the command `IPCONFIG /ALL`. You are interested in the IP Address listed after `Default Gateway`. When you have it, open your web browser and type *http://RouterIPAddress* into the address bar, but substitute the router IP address for *RouterIPAddress*. You can then login to the router. Navigate to the page which enables you to edit the Port Forwarding or Virtual Server setup. When you locate it, create a new record (if necessary) and enter the IP Address of your Raspberry Pi and the port number 80 into the record. If you are asked for a TCP or UDP or

TCP+UDP protocol option select TCP+UDP. If you are asked to specify the Port number for both Remote (WAN) and Local (LAN) port enter 80 for both.
Ensure that you save the router settings, rebooting it if necessary.

To test the router setup, and at the same check on its ability to do Hairpin Routing, you will need your External or Wide Area Network (WAN) IP Address. This is the IP address which identifies your router on the Internet at large - the "other" side of your network. One easy way to get this is to visit a website such as *http://WhatIsMyIP.com*. Make a note of your WAN IP address.
Then type the following into the address bar of your browser: *http://my-wan-ip-address* substituting your WAN IP address just found for *my-wan-ip-address*. If all is well, you should see your website as created on your Raspberry Pi! If you get an *error 404* (site not found) it could be that you have made an error in the setup or your router does not do Hairpin Routing. You can check this using a friend's Internet connection.

C3.2 Getting a Hostname and DNS

Having your site accessed by its WAN IP address is all very well, but not so easy to remember. This is why domain names are so important on the Internet, and lie at the heart of the World Wide Web. What we now need to do is to get your site a name, or URL which is linked to your WAN IP address, so that your URL can be used to access your site. This service is quite widely available free-of-charge or you can pay for a subscription service. To locate a service try a Google Search for phrases such as "*free dns hosting*". Ideally you need a service which also offers free subdomains as well. What this means is that the service provides the domain (such as *domain.com*) and you choose any available and suitable subdomain such as *my-website*. When combined this gives you a website address visible to the world as *my-website.domain.com*. After you have signed up to the service, all you have to do is to enter your WAN IP address, and chose your preferred subdomain and you have your web URL set up.

C3.3 Creating a Server Alias

Now that you have your web URL set up you will need to adjust your configuration of `Apache2` to take account of it. This is necessary because as set up currently, Apache2 will only respond to the ServerName of *oursite1*. But you want it to respond also to *my-website.domain.com*, or whatever subdomain and domain you have chosen with your DNS service.

First, stop the Apache2 service in the usual way.
Then edit your configuration file `oursites.conf` as follows:
```
      nano        oursites.conf
```
You need to add a new directive, `ServerAlias` as follows:
Locate the line:
```
      ServerName      oursite1
```
and create a new blank line immediately after it, and enter the text:
```
      ServerAlias     my-website.domain.com
```
substituting your new website URL chosen with your DNS service for `my-website.domain.com`.

If everything has been done correctly your website is now available world-wide using your newly created web URL. Your DNS service will watch for web traffic with your web address and forward it to your router using its WAN IP address. In turn, your router will send the data packets to your Raspberry Pi using its local IP address. Apache2 will pick up the data and using the web URL to identify your site by its `ServerAlias`, it will serve the requested pages back to the original client PC which could be in Outer Mongolia! And with our nest little setup you'll be able to see whether anyone in Outer Mongolia has indeed viewed your site!

Next, let's set up `Webalizer` so that you can see what work your Raspberry Pi is doing.

C4. Showing your Website Statistics

In section A12 above, you installed the `Webalizer` program. Unfortunately at present, `Webalizer` does not know about the pages being served by your newly setup web URL and its associated web site.

Apache2 is already placing data in its log files for your site *oursite1* (alias *my-website.domain.com*). See the data in the file in `/var/www/oursite1/log/access.log` by opening the file in `nano`.
In order to see the access data graphically displayed, you need to create a subdirectory for the `Webalizer` graphics and then to make two small amendments to the `Webalizer` configuration file.

First create the new directory:
```
     mkdir     /var/www/oursite1/webalizer
```
Then make the amendments to the `webalizer.conf` file by opening it in `nano`:
```
     nano      /etc/webalizer/webalizer.conf
```

First, in the `webalizer.conf` file locate and edit the `Apache2` log file entry line currently:
```
     logfile   /var/log/apache2/access.log
```
to
```
     logfile   /var/www/oursite1/log/access.log
```

Finally, locate the line beginning with the directive `OutputDir` and change it to:
```
     OutputDir /var/www/oursite1/webalizer
```

You may need to stop and start the Apache2 service for these changes to take effect.

Normally `Webalizer` processes the log files into graphics once each day. If you wish to see the effects of your changes immediately, just type the command `webalizer` and the `Apache2` statistical graphics will be updated and ready to be viewed. To view them enter the following web address into the address bar of your web browser: *http://my-website.domain.com/webalizer*.

C5. Secure your Web Statistics against prying eyes

If you wish to to keep your web statistics private, you must password protect access to the `/var/www/oursite1/webalizer` directory. To do this first navigate to the `Webalizer` directory create an encrypted `.htpasswd` file containing the username and password details:
```
     cd        /var/www/oursite1/webalizer
     htpasswd  -c .htpasswd username
```
where `username` is the username you want to use to permit access to the statistics.
You'll be prompted to enter and retype your password, then the `.htpasswd` file will be created.
Next create a new file `.htaccess` in the same directory, using `nano`:
```
     nano      .htaccess
```
and add the following lines:
```
     AuthUserFile   /var/www/oursite1/webalizer/.htpasswd
     AuthType       Basic
     AuthName       /var/www/oursite1/webalizer
     Require        valid-user
```
Remember to adjust the parameters to suit your own site name and directories!

When you next access your `Webalizer` web statistics directory in your browser, you will be asked for the username and password!