Basingstoke MakerSpace

Implementing Pi-hole

a network wide ad blocker

Introduction

The Pi-hole® is a DNS sinkhole that protects all your devices from unwanted content, without installing any client-side software. The site at https://pi-hole.net/ provides everything you need to implement the solution, this document supplements that information by providing a more detailed guide. It was put together by Steve Hawtin to accompany some interactive sessions run by the Basingstoke MakerSpace (www.basingstokemakerspace.org.uk/). This document has not been approved or validated by the Pi-hole project in any way.

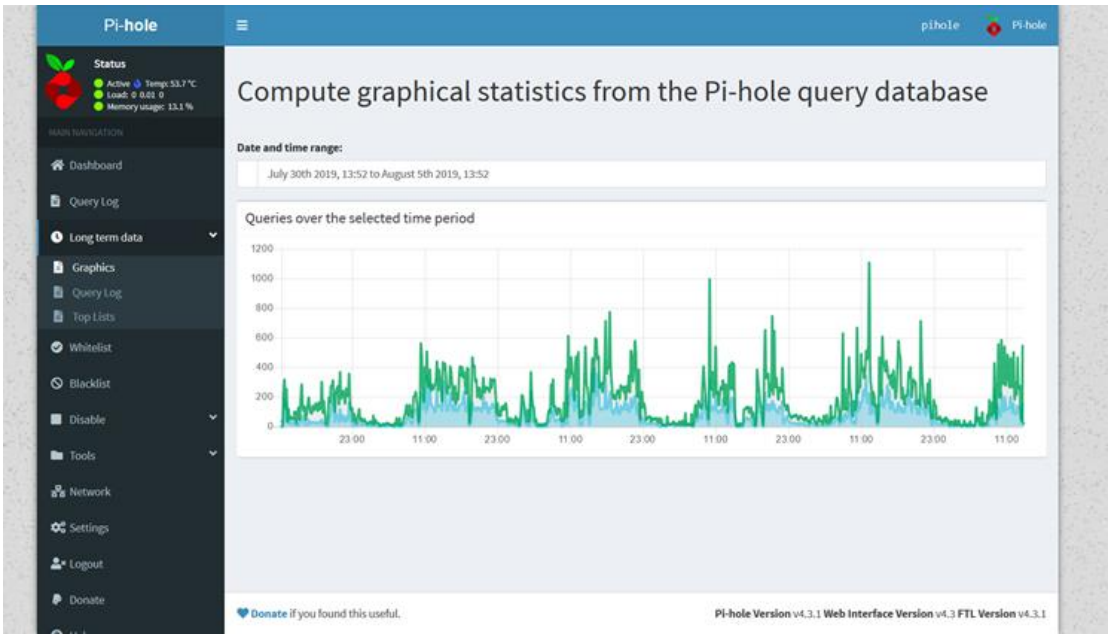

Figure 1 The usage curve of a running Pi-hole server

Overview

In order to understand how Pi-hole works it is important to appreciate two major steps that occur when a computer requests a web page.

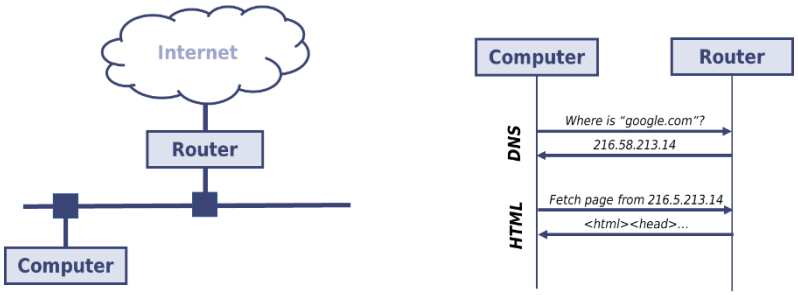

Figure 2 In most houses the router handles all the network admin tasks

In most houses there is an internal network (usually a mix of physical wires and Wi-Fi) that connects the various computers to the internet. When a web page is requested the name of the web site is translated into an IP address and a request is sent to that address to obtain the page contents. The translation from a text name to an IP address is performed by DNS (Domain Name Service), fetching the page contents is HTTP (HyperText Transfer Process). In most houses these two distinct steps are both handled by the local router. If the web page contents incorporate elements from elsewhere then the browser goes through the whole process again for these sub elements. Because this is a recursive process a single web page can potentially lead to many sub-requests (for example as adverts are fetched).

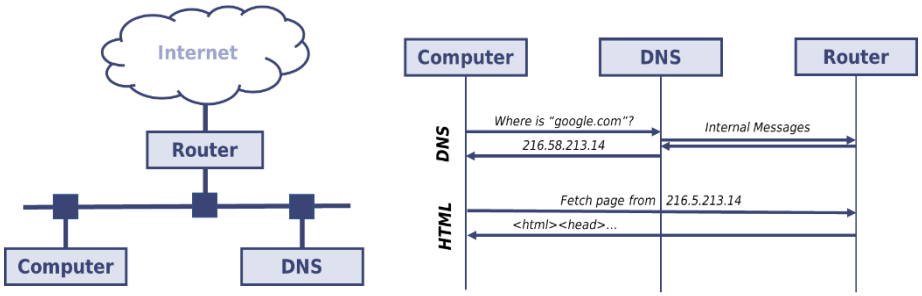

Figure 3 A separate DNS server allows control over which domains are translated

The Pi-hole implements a DNS server in a separate machine. When fetching a web page the computer asks that server to translate the name and uses its reply to request the web page contents. On the face of it this might seem like a rather stupid idea. After all the DNS server is just going to have to go to via the router to resolve the name anyway. There are two different reasons why this simple view is wrong, first of all the DNS server can remember old queries and does not have to redo them (this is called caching). More importantly the DNS server can work out which queries are related to components we don't want (like adverts) and can just pretend that those names can't be found.

So, in summary, Pi-hole provides a dedicated machine to just reply to DNS requests.

Required hardware

In order to implement a Pi-hole you will require:

- A Raspberry Pi. Any version would do
- A suitable power supply
- A 16Gb micro-SD card (or SD card if you have a very early Raspberry Pi)

I would connect the Pi-hole with an Ethernet cable, this makes things easier if you have to restart the whole system (for example after a power cut). If you decide to connect using WiFi you will be able to configure that on your own.

Install Raspbian

The latest version of Raspbian, this can be found at **https://www.raspberrypi.org/downloads/** Since this is a dedicated server I would always download the lite version (the one with no graphical interface). Personally I use an out of date tool called "**imageLoader**" to place the downloaded image onto a microSD card. If you don't have a preferred tool you should probably use "**balenaEtcher**".

balenaEtcher is able to install straight from the zip file. Other tools may require that it is unpacked, tools like winzip and Gzip will be able to do that but the zip that comes with Windows will fail if the target file is bigger than 4Gb.

In theory you can configure a Raspberry Pi by writing files onto the boot partition before first boot. In practice this has never worked for me, I always plug in a keyboard and TV (via the HDMI) and configure on the screen. So, we install the microSD card, plug in the Ethernet cable, the HDMI, the keyboard on a power supply. After a while we should get a login prompt. Login with the user **pi** and password **raspberry**.

Then there is a standard sequence of things to do:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo raspi-config
```

The raspi-config too is used to:

- Resize the partitions (if necessary)
- Change the **pi** account password
- Change the machine's hostname (under **Network Options**)
- Enable **ssh** access (under **Interfacing Options**)

When you quit from that tool don't reboot. Before disconnecting the monitor we need to make a note of the IP address

```
pi@parrot:~ ifconfig
...
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
```

```
    inet 192.168.186.215  netmask 255.255.255.0  broadcast 192.168.
        inet6 fe80::4bc3:d55a:a85e:99fe  prefixlen 64  scopeid 0x20
        ether b8:27:eb:90:25:ec  txqueuelen 1000  (Ethernet)
        RX packets 6107731  bytes 194592042 (185.5 MiB)
        RX errors 0  dropped 26  overruns 0  frame 0
        TX packets 3061629  bytes 208222552 (198.5 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
...
```

Once we have the IP address we can turn off the Pi (with the command "**sudo shutdown -h now**").

Now the Pi can be moved to where it will be positioned long term (under the desk in the study?). It will need an Ethernet cable and a power supply.

You can now log on to the Pi from your main computer via ssh, for example you can use the PuTTY command (which may need to be installed).

Configure Pi-hole

The next step is to set up the Pi-hole software. The command to do this is:

```
curl -sSL https://install.pi-hole.net | sudo bash
```

(Note the extra bit in blue that is missing from the normal web descriptions). This way of configuring has the disadvantage that you don't get to see all the details, so if you are a careful person you should do the individual steps. However it has the great advantage that you don't get to see all the details, so if you are not paranoid it is quicker.

Note the admin password that you are provided with at the end of this process.

Set up your Router

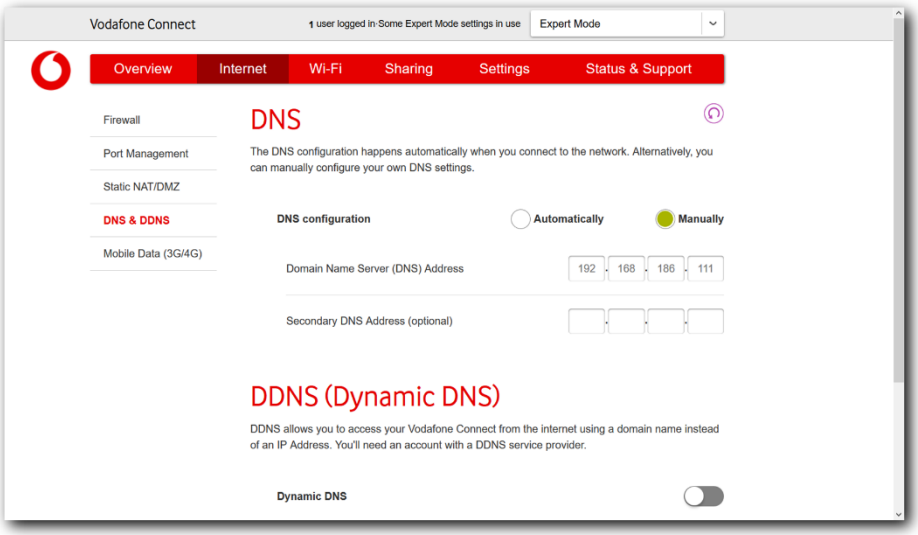

Figure 4 The control for setting up DNS on the Vodafone router

Each router employs a different mechanism for configuring the DNS server to use. For example the figure above shows how this is done on a Vodafone router. The IP address of the Pi-hole (the one we noted during configuration) has been inserted here as the DNS. You will just have to log into your router's control page and find the place where the DNS is defined.

A comprehensive description of how to set up your network the use Pi-hole can be found at:

**https://discourse.pi-hole.net/t/how-do-i-configure-my-devices-\
to-use-pi-hole-as-their-dns-server/245**

If you are really unlucky you may find that the router does not provide anywhere for you to set the DNS. In that case one option is to turn off the router's DHCP service instead install the ISC DHCP server on the Raspberry Pi. There are a whole load of tutorials on the web that you can follow.

Results

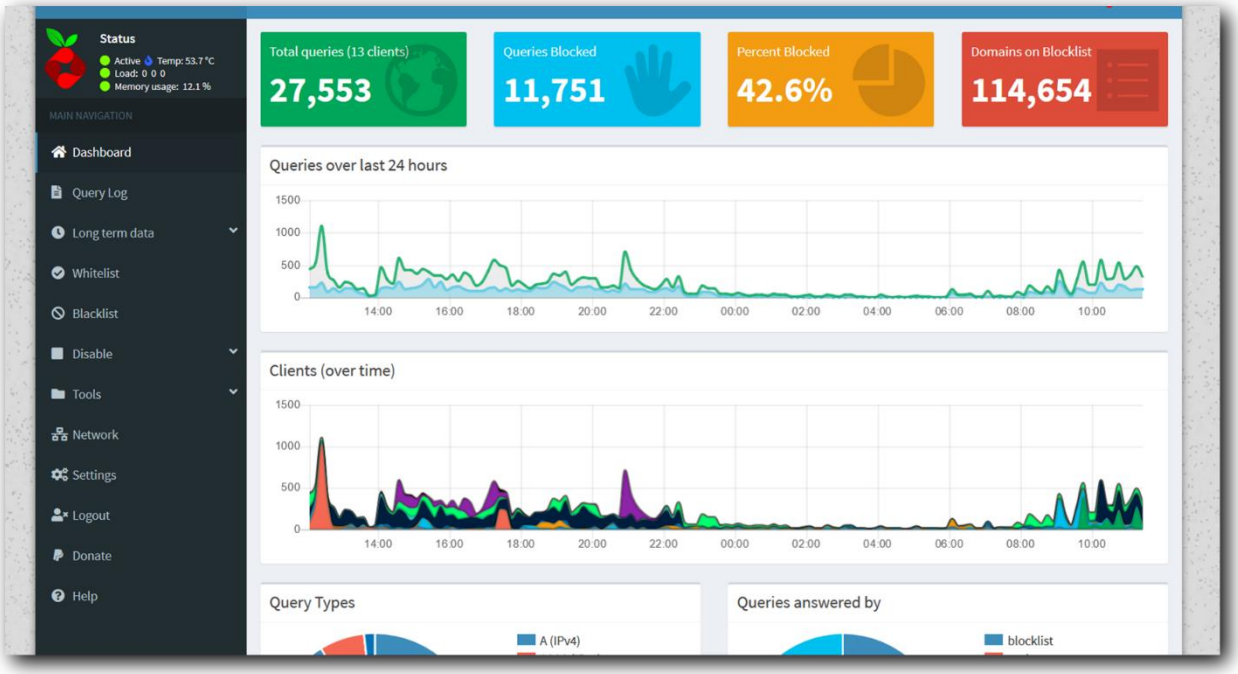

Figure 5 The dashboard of a running Pi-hole

When the server has been running a while you can inspect its progress using a web browser. Point the browser at the IP address (you will need to add "`/admin/`" to get to the dashboard). It will show you something like the page above. In this case we can see that this Pi-hole has blocked more than 40% of all the queries.