

2025

Block Ads Network-wide with A Raspberry Pi-hole



<https://discourse.pi-hole.net/u/jpgpi250>

30-3-2025

1.	About this manual.	3
2.	Buy.	3
3.	Download.	3
4.	Raspberry pi Installation.....	4
1.	Preparing the SD card.....	4
2.	Preparing your DHCP server.....	4
3.	Power on the Raspberry pi.....	5
4.	Use Putty on your (Windows) workstation to connect to the Raspberry pi.	5
5.	Resize the Linux partition.	5
6.	Configure the static IP address.....	5
7.	Update the Raspberry pi.	5
8.	Reboot the Raspberry pi.....	5
9.	Repair the update (if a kernel patch has been installed).	6
10.	Install mail (assumes valid gmail account).	6
11.	Setup Key authentication	7
12.	Install Webmin (version 2.303).	9
13.	Additional system configuration (webmin).	10
14.	Time (timesyncd) configuration.	11
15.	Install DNS utils.....	12
5.	Pi-hole installation (version v6.0.6).....	13
1.	What will pi-hole do for you?	13
2.	Installation.....	13
3.	Upgrading.	14
4.	Pi-hole configuration.	15
5.	Time (built-in NTP server) configuration.....	16
6.	Change your DNS settings.	17
7.	Change the default UNIX password.	17
8.	Change / Recover the admin page password.....	17
9.	Adding a local LAN list	18
10.	Windows Whitelist.	19
11.	Modify lists using the 'pihole' command.	20
12.	Modify lists using sqlite3.	21
13.	Adding Wildcard sites to the blacklist.	21
14.	Regular expressions.....	22
15.	Deep CNAME inspection	23
16.	Adding host lists.	24

1.	Block Lists.	24
2.	Adblock Plus domain lists.	26
3.	Subscribed allow lists (antigravity).....	27
17.	Group Management	28
1.	Groups.	28
2.	Clients.....	28
3.	Domains.....	29
4.	Adlists	29
18.	Group Management, a whitelist example.....	29
19.	Group Management, duplicate blacklist and whitelist entries	30
20.	Windows DNS cache.....	31
21.	Changing the hostname.	32
22.	Protect your Raspberry Pi.....	32
23.	Disable unused hardware (Raspberry Pi® 3B, 3B+, 3A+, 4B and Zero W).	33
1.	Disable WIFI.....	33
2.	Disable Bluetooth.	33
24.	Using cron.....	33
1.	Add new NextDNS entries (weekly).	33
2.	Run needrestart (daily).....	33
25.	Install some useful system tools	34
1.	Install watchdog	34
2.	Install schedtool	35
3.	Install needrestart.	35
4.	Helping the RANDOM number generator.	36
26.	IPv6 address.....	36
27.	Backup your pi-hole.....	37
28.	Customizing pi-hole (optional).	38
1.	PHP info.	39
2.	Lighttpd Server Status.	39
3.	Browsing the FTL database.....	40
29.	DoT / DoH	41
30.	Using unbound as upstream resolver.	42
31.	Change Log	42

1. About this manual.

Using this guide will turn your Raspberry pi into a secure, manageable and stable pi-hole system.

If you are reading this document, using Adobe Reader, you may click on a hyperlink to content in this document. Use the combination <Alt> <left arrow> to return to the previous location.

"Back" and "Forward" buttons can also be added to the toolbar. If you right-click on the tool bar, under "Page Navigation", they are referred to as "Previous View" and "Next View".

This document is hosted on [GitHub](#), you can open the document (pdf), using this [link](#).

Copying and pasting from this manual into [Putty](#) doesn't seem to work all the time. If you get an error, try typing the command...

2. Buy.

You can buy this anywhere, I bought them at Conrad (included links). If you buy them at Conrad, ensure you use the country specific links ([conrad.de](#), [conrad.be](#), [conrad.nl](#) ...), this to get the proper payment and delivery options!

- Raspberry pi:
 - o Raspberry Pi® 3 Model B 1 GB w/o OS (10/100 Ethernet - item no.: [1419716](#)) **OR** Raspberry Pi® 3 Model B+ 1 GB w/o OS (Gigabit Ethernet - item no.: [1668026](#))
 - o Banana Pi® B+ enclosure Black RB-Case (item no.: [1274195](#))
- SD card: Ensure you buy a class 10 card. You'll need an SD adapter to format and write the SD card.
 - o microSDHC card 32 GB Transcend 32GB CL10 MICRO SDHC CARD Class 10 (item no.: [416521](#))
 - o Transcend MicroSD™ Adapter auf SD (itm no.: [1413689](#))
- Power Supply: If you don't have a spare one.
 - o power supply unit Black RB-Netzteil3-B (item no.: [1429556](#))

3. Download.

- [Putty](#), ensure you download a version including PuTTYgen.
- [WinSCP](#)
- [Win32DiskImager](#)
- [SDFormatter](#)
- [Raspberry Pi OS \(32-bit\) Lite](#) (previously called Raspbian). This document was written, using the November 19th 2024 [release](#), Kernel version 6.6. If your hardware is compatible (see compatibility list on the [download page](#)), you might consider using the 64-bit version.

If you are using the 64-bit version:

- The OS will be different (32-bit: raspbian, 64-bit:debian), ref. /etc/os-release
- This manual was written, using the 32-bit version, there might be small differences when using the 64-bit version.
- Be aware you may need to [increase the systems virtual memory size](#), when using a 64-bit OS. The size depends on the additional packages installed, use **free -mh** to monitor usage, increase if required (default is 100Mb).

- [MD5 & SHA Checksum utility](#). Whenever possible, verify the integrity of your download, using the provided checksum.
- Optional: [DiskGenius](#), SD card analysis.

4. Raspberry pi Installation.

1. Preparing the SD card.

- Format the SD card, using SDFormatter.
- Extract 2024-11-19-raspbian-bookworm-armhf-lite.img.xz, this zip contains a single image file.
- Write the extracted file (2024-11-19-raspbian-bookworm-armhf-lite.img) to the SD card, using Win32DiskImager.
- You need to create a file called “ssh” (**no extension**) in the boot partition to enable SSH (read the [release notes](#)).



- You need to create a file called “userconf.txt” (single line, no additional linefeed) in the boot partition to create the default user (read [here](#)), content (user **pi**, password **raspberrypi**, as in previous images). Read the instructions in the news item to generate a different initial password.

```
1 pi:$6$/4.VdYgDm7RJ0qM1$FwXCeQgDKkqrOU3RIRuDSKpauAbBvP11msq9X58c8Que2l1Dwq3vdJMgiZlQSbEXGaY5esVHGBNbCxKLVNqZW1
```

Encrypted password:

\$6\$/4.VdYgDm7RJ0qM1\$FwXCeQgDKkqrOU3RIRuDSKpauAbBvP11msq9X58c8Que2l1Dwq3vdJMgiZlQSbEXGaY5esVHGBNbCxKLVNqZW1

- I’ve tried several solutions (Win32DiskImager, Etcher) to backup and restore (clone) the SD card, unfortunately, they all fail, due to the “Not enough space error”. Most internet search results recommend to shrink the size of the Linux partition, before creating the image, this to allow successful cloning, an additional Linux machine is often required. **If you want to successfully backup / restore (clone) your SD card**, you may want to execute the steps, described in [this document](#), before you start building your pi-hole.
- Insert the SD card in the Raspberry pi (power disconnected).

2. Preparing your DHCP server.

You probably have an existing DHCP configuration. It is advised you make a static entry for the Raspberry pi (IP address – MAC address). This will ensure the Putty and winSCP

configurations will still be functional, if you decide to reinstall from scratch. The [static IP](#) configuration will overwrite the values from the DHCP server.

3. Power on the Raspberry pi.

You only need to connect the power and an Ethernet cable. There is no need for a keyboard, mouse or HDMI monitor.

4. Use Putty on your (Windows) workstation to connect to the Raspberry pi.

- Session / Host Name (or IP address): enter the IP address
- Connection / Data / Auto-login username: pi
- Session / Saved Sessions: Enter a name for the device and click 'Save'
- Click 'Open'
- The default password is '**raspberrypi**'

5. Resize the Linux partition.

If you disabled automatic partition expansion, you need to resize the partition, following the instructions in [this document](#), this to allow for successful [backup](#) / restore (clone).

6. Configure the static IP address.

The latest version of the OS (bookworm) uses NetworkManager instead of dhcpcd as networking interface. In order to configure a static IP and force the OS to use specific DNS servers, you can use **nmcli**, ref [here](#).

```
sudo nmcli con mod "Wired connection 1" \  
ipv4.addresses "<your Raspberry pi's static address>/<netmask>" \  
ipv4.gateway "<your networks gateway>" \  
ipv4.dns "208.67.222.222,208.67.220.220" \  
ipv4.method "manual"
```

Use the multiline method (enter a backslash (\) after every line) or enter everything on a single line (without the backslashes)

Now is the time, if you haven't already done so, to configure the static DHCP entry. To find the MAC address:

```
ifconfig
```

Copy the **HWaddr** (that is the MAC address) from **eth0**

7. Update the Raspberry pi.

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

Wait for the updates to install...

8. Reboot the Raspberry pi.

This is required to activate the static IP address and possible Raspberry pi specific patches to the Linux kernel.

Your Putty session will disconnect, wait a few seconds, right click the Putty frame and select 'restart session'.

```
sudo reboot
```

9. Repair the update (if a kernel patch has been installed).

If a Linux kernel patch has been installed, you need to issue the following commands to complete/repair the raspbian update:

```
sudo apt-get update  
sudo apt-get -y --fix-broken install  
sudo apt-get -y autoremove
```

10. Install mail (assumes valid gmail account).

Using another service than gmail may require specific configuration settings.

Reference: <https://websistent.com/how-to-use-msmtp-with-gmail-yahoo-and-php-mail/>

Reference: <https://websistent.com/msmtp-cron/>

We will be installing MSMTTP, you will need to [upgrade](#) the system, if you haven't already done so, before this works!

```
sudo apt-get -y install msmtp
```

Wait for the installation to complete...

Create a symbolic link for sendmail (required for [webmin](#), see reference)

```
sudo ln -s /usr/bin/msmtp /usr/sbin/sendmail
```

Create the MSMTTP configuration:

```
sudo nano /etc/msmtprc
```

Google changed the [policy](#) for less secure apps. As a result you need to generate an app password, and use this password in your configuration. In order to activate this option (read [here](#) how to generate an app password), 2FA is required. Ensure you have sufficient 2FA recovery options (device, phone, google authenticator app, backup codes).

Replace <your account name> (twice) with a valid gmail address and update <your app password>.

```
defaults  
tls on  
auth on  
host smtp.gmail.com
```

```
port 587
user <your account name>
password <your app password>
aliases /etc/aliases
account default
from <your account name>
```

Edit the MSMTTP aliases configuration.

```
sudo nano /etc/aliases
```

Add the following (replace the account information)

```
default: <your account name>
```

Optionally, send a test mail, you can read more about this [here](#), this simple test sends a mail without a subject (replace the account information):

```
echo "msmtp test mail" | msmtp -a default <your account name>
```

You also will get some mails, while making changes to the [webmin](#) configuration.

11. Setup Key authentication

Generate the authentication keys on your Raspberry pi

```
ssh-keygen -t rsa -C "raspberrypi"
```

Accept the defaults

If you didn't already setup WinSCP on your (Windows) workstation:

- Open WinSCP, select 'New Site'
- File protocol: SCP
- Host name: <your Raspberry pi's static address>
- User name: pi
- Password: raspberry
- Click 'Advanced'
- Environment / SCP/shell /Shell: sudo su –
- Click "OK"
- Click "Save"

Login, using WinSCP

- Select the saved session
- Click "Login"
- Select Options / Preferences from the WinSCP menu
- Select Environment / Interface

- Check Commander
- Select Panels
- Check Show hidden files

Browse to the pi .ssh directory (/home/pi/.ssh)

Copy id_rsa and id_rsa.pub to your (Windows) workstation (It's recommended you create a sources/installation/key folder for your Raspberry Pi, containing all the necessary files)

Rename id_rsa.pub to **authorized_keys** (no extension) and copy it back to the .ssh folder. If you want to restrict SSH logins to particular IP addresses, check out this [reference](#).

Start [PuTTYgen](#) on your (Windows) workstation.

- Select "Load"
- Select the "All files" type
- Browse to your sources/installation/key folder and select id_rsa
- Click "Open", Confirm the import
- Click "Save private key"
- Confirm you want to save the key without a passphrase
- Type an appropriate key name and save the private key file (.ppk)

Configure Putty to use the key

- Open Putty, select the saved session, click "Load"
- Connection / Data / Auto-login username: pi
- Connection / SSH / Auth / Credentials
- Click "Browse", select the private key file you created (.ppk)
- Session
- Click "Save"

Test your configuration, open a new Putty session, you should be logged on automatically.

Configure WinSCP to use the key

- Open WinSCP, select the saved session, click "Edit"
- Click "Advanced"
- SSH / Authentication
- Private key file
- Click "..." (Browse), select the private key file you created (.ppk)
- Click "OK" (closes advanced)
- Empty the password field
- Click "Save"

Test your configuration, open a new WinSCP session, you should connect, using the private key.

In order to ensure key security, apply the following:

```
sudo chown pi:pi /home/pi/.ssh/authorized_keys
```

```
sudo chmod 600 /home/pi/.ssh/authorized_keys
sudo chown pi:pi /home/pi/.ssh/id_rsa
sudo chmod 600 /home/pi/.ssh/id_rsa
sudo chown pi:pi /home/pi/.ssh/id_rsa.pub
sudo chmod 644 /home/pi/.ssh/id_rsa.pub
```

Further increase security by adding the IP address of your workstation(s):

```
sudo nano /home/pi/.ssh/authorized_keys
```

This file contains the key, used to allow authentication.

Insert the following at the beginning of the line (before the key) to add the IP address limitation (replace the IP address with the IP address of your workstation). A space after the last double quote is required:

```
from="192.168.x.x"
```

Specifying multiple IP address is an option (enter multiple IP addresses, allowed to use SSH):

```
from="192.168.x.x,192.168.x.y"
```

12. Install Webmin (version 2.303).

Reference: <http://www.webmin.com/deb.html>

Install the dependencies, some packages, such as python, may already be installed.

```
pi@raspberrypi:~ $ which python
/usr/bin/python
pi@raspberrypi:~ $ python --version
Python 3.11.2
```

```
sudo apt-get -y install apt-show-versions
sudo apt-get -y install html2text
sudo apt-get -y install libauthen-pam-perl
sudo apt-get -y install libcommon-sense-perl
sudo apt-get -y install libdbd-mysql-perl
sudo apt-get -y install libdbi-perl
sudo apt-get -y install libio-pty-perl
sudo apt-get -y install libjson-xs-perl
sudo apt-get -y install libnet-ssleay-perl
```

```
sudo apt-get -y install libpam-runtime
sudo apt-get -y install libtypes-serialiser-perl
sudo apt-get -y install openssl
sudo apt-get -y install perl
sudo apt-get -y install python3
sudo apt-get -y install shared-mime-info
```

Download the package.

```
sudo wget http://prdownloads.sourceforge.net/webadmin/webmin_2.303_all.deb
```

Install the package, this may take a while...

```
sudo dpkg --install webmin_2.303_all.deb
```

13. Additional system configuration (webmin).

The Webmin URL: <https://<Your Raspberry pi's IP address>:10000/>

The username is pi, the password is raspberry, unless you've already [changed](#) that.

- Webmin / Webmin configuration / Logging:
 - o Requires [mail setup](#)!
 - o Modules to log Webmin actions in: Select **Only log actions in ..**
 - Select (CTRL click) **Software Package Updates**
 - Select (CTRL click) **Webmin Configuration**Failing to make a selection (leaving the setting to **Log actions in all modules**) will cause a mail storm from the **Webmin Scheduled actions**
- System / Software Package Updates / Scheduled Upgrades:
 - o Requires [mail setup](#)!
 - o Check for updates on schedule: **Yes**, every **day**.
 - o Email updates report to: enter a valid (g)mail address.
 - o Action when update needed: **Install any updates**.

The webmin configuration screen doesn't allow you to set the time for scheduled updates. You can change the time, using the "Edit Cron Job" editor (system / scheduled cron jobs / click on "/etc/webmin/package-updates/update.pl"), make sure it is scheduled to run before your [needrestart](#) script will run (allow sufficient time to complete large updates).

- Hardware / system time / change timezone:
 - o Select the correct time zone

If you know the name of your time zone, you can also change it on the command line. Example for "Europe/Brussels":

```
sudo timedatectl set-timezone Europe/Brussels
```

- Webmin / Webmin Configuration / IP Access Control:
 - o Select “only allow from listed addresses”
 - o Enter allowed IP addresses (at least the **static** IP address of your workstation)
- Servers / SSH Server / Authentication:
 - o Requires working [Key authentication](#)!
 - o Allow authentication by password? **No**

Need more info? The manual (by Jamie Cameron) can be found [here](#), Webmin issues [here](#) (GitHub).

14. Time (timesyncd) configuration.

The latest versions of some linux distributions, including Raspbian, come with a service, called ‘systemd-timesyncd.service’. This minimalistic service (not my words, see the [man page](#)) may be used to synchronize the local system clock. It also saves the local time to disk (/var/lib/systemd/clock) every time the clock has been synchronized. If you don’t mind the additional writes to your SD card, you may use this service to keep time synchronized, you will need to configure the service.

To check if the service is available:

```
sudo service systemd-timesyncd status
```

On a fresh installed Raspbian system, this will show ‘active (running)’

To configure the service:

```
sudo nano /etc/systemd/timesyncd.conf
```

Modify the line ‘#NTP=’. It is recommended to change the server value (“0.debian.pool.ntp.org”) into the name (or IP address) of the NTP server, as provided by your ISP.

```
NTP=<name or IP address of NTP server, multiple entries separated by space>
```

Restart the ‘systemd-timesyncd’ service

```
sudo systemctl daemon-reload
sudo service systemd-timesyncd restart
```

To verify time synchronization, enter

```
timedatectl
```

The reply should indicate that the System clock is synchronized.

If you are happy with using the ‘systemd-timesyncd’ service, move on to the next section ([install DNS utils](#)).

If you intend to use the NTP service built into pi-hole-FTL (see [here](#)), move on to the next section ([install DNS utils](#)).

If you prefer to use a more robust time service, continue.

A package, available for almost all linux distributions is [chrony](#). This package installs a daemon that will keep time synchronized.

The chrony server package isn't installed by default in this version of Raspbian.

Install the chrony package:

```
sudo apt-get install chrony
```

Goto <https://support.ntp.org/bin/view/Servers/StratumTwoTimeServers>

Select the region you are in, there will be a list of NTP servers for your region. Select at least two servers, use IP addresses to eliminate DNS resolver dependence. Alternatively, you might want to use your providers NTP servers.

Now add the servers to the configuration file (replace the IP addresses – use append for the second server entry), checkout the README file in /etc/chrony/sources.d/:

```
echo 'server xxx.xxx.xxx.xxx iburst' | sudo tee /etc/chrony/sources.d/local-ntp-server.sources  
echo 'server yyy.yyy.yyy.yyy iburst' | sudo tee -a /etc/chrony/sources.d/local-ntp-server.sources
```

Activate the changes:

```
sudo chronyc reload sources
```

Stop the 'systemd-timesyncd' service (the minimalistic NTP service that might be running on your system and disable it permanently (you can enable it again if you change your mind).

```
sudo service systemd-timesyncd stop  
sudo systemctl disable systemd-timesyncd
```

Check time server synchronization status.

```
chronyc sources
```

You'll get a list of servers, the primary server is marked with an asterisk (*). It may take a while for the synchronization to become active, repeat the command

15. Install DNS utils.

It is recommended to check your system's DNS capability before installing pi-hole.

```
sudo apt-get -y install dnsmasq
```

Check if name resolution is functional, remember we configured the [OpenDNS](#) servers.

```
dig google.com
```

5. Pi-hole installation (version v6.0.6).

1. What will pi-hole do for you?

Pi-hole will provide an answer for all domain queries, to any device that uses pi-hole as DNS server. The answer can be the real address for the domain or an answer that will prevent the device loading content from this domain. The format of the answer can be configured, by [changing](#) the setting 'dns.blocking.mode = "NULL"'. The blocking mode options can be found [here](#). Pi-hole uses 'NULL' blocking (default and recommended).

```
Mar 3 11:48:48 dnsmasq[5851]: query[A] collector-hpn.ghostery.net from 192.168.2.228
Mar 3 11:48:48 dnsmasq[5851]: forwarded collector-hpn.ghostery.net to fdaa:bbcc:ddee:2::5552
Mar 3 11:48:48 dnsmasq[5851]: query[AAAA] collector-hpn.ghostery.net from 192.168.2.228
Mar 3 11:48:48 dnsmasq[5851]: forwarded collector-hpn.ghostery.net to fdaa:bbcc:ddee:2::5552
Mar 3 11:48:48 dnsmasq[5851]: reply collector-hpn.ghostery.net is <CNAME>
Mar 3 11:48:48 dnsmasq[5851]: reply collector-hpn.privacy.ghostery.net is 54.194.146.104
Mar 3 11:48:48 dnsmasq[5851]: reply collector-hpn.privacy.ghostery.net is 52.45.85.77
Mar 3 11:48:48 dnsmasq[5851]: reply collector-hpn.privacy.ghostery.net is 54.86.217.191
Mar 3 11:48:48 dnsmasq[5851]: reply collector-hpn.ghostery.net is <CNAME>
Mar 3 11:48:48 dnsmasq[5851]: reply collector-hpn.privacy.ghostery.net is NODATA-IPV6
```

Click to play, using sample data

The data, used in the demo, is extracted from an operational system, using numerous [block lists](#). All blocked entries are displayed in **magenta**, resulting in an unspecified IP.

2. Installation.

Reference: <https://pi-hole.net/>

- **Optional:** If you will be hosting other services on your raspberry pi, which need to be available on port 80, it is easier to install the service prior to installing pi-hole. In order to look at the content of the pi-hole FTL sqlite3 database, see "[Browsing the FTL database](#)", you'll need to install lighttpd (or another suitable web server) with php enabled.

By installing this service, prior to pi-hole, the installation process will automatically select an alternative port (8080) for the administration web site.

In order to install lighttpd (web server), run:

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

If lighttpd is installed prior to pi-hole, the "webserver.port" setting (settings explained [here](#)) will be changed into:

Value (string)

```
8080,443s,[::]:8080,[::]:443s
```

Default Value: "80,[::]:80,443s,[::]:443s"

Pi-hole also comes with a version of sqlite3 embedded in pihole-FTL, however, if you will be installing phpliteadmin (see "[Browsing the FTL database](#)") the package "sqlite3" is also required, thus, optional, install sqlite3:

```
sudo apt-get -y install sqlite3
```

- Automated install

I've had issues with this (DNS error) see below for an alternative

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

- Alternative Semi-Automated install

```
wget -O basic-install.sh https://install.pi-hole.net
chmod +x basic-install.sh
sudo ./basic-install.sh
```

- Read the informational dialogs, select **"Continue"** when the **"Static IP Needed"** dialog appears.
- Select DNS servers (I've been using the OpenDNS servers), **"OK"** to continue.
- Select **"Yes"** to include the third party list (StevenBlack).
- Select **"Yes"** to log queries. Your choice will be recorded, using the "queryLogging" setting in /etc/pihole/pihole.toml.
- Select a privacy mode for FTL that suits you, defaults to 0 (Show everything), recommended, select **"Continue"**.
- Select **"No"** when asked to disable lighttpd, assuming you opted to install lighttpd.
- Wait for the installation to complete...
- Write down the web interface url. Notice the IPv6 address, listed in the dialog. If the address is incorrect, you may experience strange behavior. Check the [forum IPv6 topics](#) for a solution.
- Don't forget to configure the correct [DNS settings](#)...

3. Upgrading.

You may notice a message "Update available!"

Core v6.0 · **Update available!** **FTL v6.0** **Web interface v6.0**

To install updates, run `pihole -up`.

To find your pi-hole version

```
pihole version
```

You can upgrade using the command

```
pihole updatePihole
```

You can automatically install updates (**not recommended**), if any. You'll need to add a [cron job](#).

```
sudo nano /etc/cron.d/piholeupdate
```

Add the update job by adding the following, modify the time to meet your requirements.

```
# Pi-hole: Update Pi-hole
30 2 * * 7 root PATH="$PATH:/usr/local/bin/" pihole updatePihole
```


4. Pi-hole configuration.

There are several options to change the pi-hole configuration. You can modify pi-hole to suit your needs, using:

- The Settings Menu (web interface): Clicking on any “Settings” sub menu will show the “Settings level” in the upper right corner (“Basic”, “Expert”). Changing the level to “Expert” will reveal an additional menu choice “All Settings”, which offers the level selection “All settings” and “Modified settings”. Explaining all of these settings is out of scope, refer to the pi-hole documentation for more details.

On the “All Settings” sub menu, a setting that has been changed (no longer default) is tagged with an icon, example:

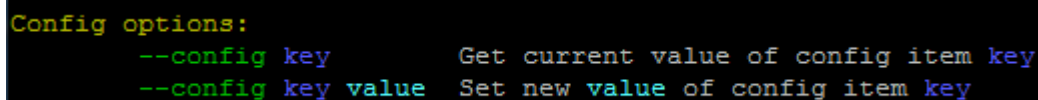


dns.upstreams  
Array of upstream DNS servers used by Pi-hole
Example: ["8.8.8.8", "127.0.0.1#5353", "docker-resolver"]

- Settings can be modified by editing the settings file (/etc/pihole/pihole.toml, this requires the pihole-FTL service to be stopped, in order to save the changes).

```
sudo nano /etc/pihole.toml
```

- Settings can be viewed and modified on the command line, using CLI. The pihole-FTL help message (pihole-FTL --help) shows:



```
Config options:
--config key      Get current value of config item key
--config key value Set new value of config item key
```

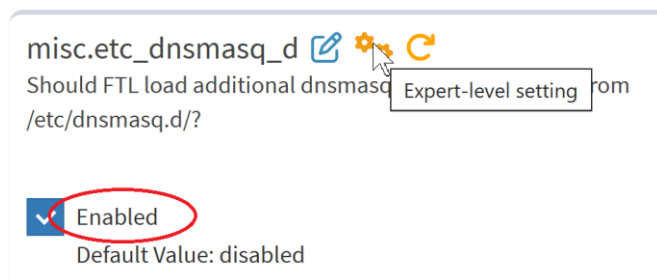
Example: show the current blocking mode:

```
pihole-FTL --config dns.blocking.mode
```

Example: change the blocking mode to IP blocking:

```
sudo pihole-FTL --config dns.blocking.mode IP
```


The pi-hole configuration file (/etc/pihole/pihole.toml) is used to generate a dnsmasq configuration file (/etc/pihole/dnsmasq.conf). Don't change this file, it is recreated when pi-hole-FTL starts. In order to change the default behavior to suit your needs, you can use additional dnsmasq settings. If you only need to add a few dnsmasq settings, add them to "misc.dnsmasq_lines", using the web interface (or edit the configuration file and restart pi-hole-FTL). The more convenient way for expert users is "misc.etc_dnsmasq_d". Changing this setting will allow you to add configuration files, using the additional configuration options of dnsmasq, see the [man page](#). The additional files, stored in **/etc/dnsmasq.d**, will not be processed by pi-hole-FTL, if the setting isn't enabled (default value: Disabled).



There are numerous topics in the forum that will change pi-hole's behavior, using additional dnsmasq settings, at least one of the above settings ("misc.dnsmasq_lines" and/or "misc.etc_dnsmasq_d") needs to be used/enabled for these suggestions to work.

5. Time (built-in NTP server) configuration.

Version 6 of pi-hole comes with NTP (network time protocol) built into pi-hole-FTL. Since you already configured the NTP client service, it isn't necessary to enable the pi-hole-FTL NTP services (default enabled), you may want to change the relevant settings to match your preferences.

The relevant settings are:

```
[ntp.ipv4]
  active (default true)
  address (default "")

[ntp.ipv6]
  active (default true)
  address (default "")

[ntp.sync]
  Active (default true)
  server (default "pool.ntp.org")
  interval (default 3600)
  count (default 8)

[ntp.sync.rtc]
  set (default true)
  device (default "")
  utc (default true)
```

You can change these settings (see [configuration](#)), once pi-hole is installed. It is recommended to change the server value ("pool.ntp.org") into the name (or IP address) of the NTP server, as provided by your ISP. The entry should match the entry you configured [here](#).

If you don't want to use the built-in NTP service, you can disable it by changing the values of `ntp.ipv4.active` and `ntp.ipv6.active` to false, to disable the client, change the value of `ntp.sync.interval` to zero (0) or `ntp.sync.server` to "" (empty string), reference:



Failed to adjust time during NTP sync: Insufficient permissions

you can easily disable this feature by seeing either the config option `ntp.sync.interval = 0` or `ntp.sync.server = ""`.

If you are using the built-in NTP service, you can stop the 'systemd-timesyncd' service (the minimalistic NTP service that might be running on your system and disable it permanently (you can enable it again if you change your mind).

```
sudo service systemd-timesyncd stop  
sudo systemctl disable systemd-timesyncd
```

6. Change your DNS settings.

Pi-hole won't do anything, unless you modify the DNS settings on your (Windows) workstation(s).

If you have a DHCP server on your network, change the DNS settings in DHCP server setup. The first DNS server should be <Your Raspberry pi's IP address>. Don't configure a second DNS server, unless you have two pi-holes. You'll need to reboot your workstation for the new DNS setting to become active immediately.

If you're using a local DNS configuration, you'll have to change it on all the devices.

You'll also need to flush or [configure the DNS cache](#) on your (Windows) workstation.

```
ipconfig /flushdns
```

7. Change the default UNIX password.

The default password for the pi user is raspberry. In order to protect the system, you need to change this. We're using sudo to allow simple passwords. [Webmin](#) will also be accessible, using the new password.

```
sudo passwd pi
```

Enter the new password.

8. Change / Recover the admin page password.

You can change the admin page password, using putty.

Enter the following command:

```
sudo pihole setpassword
```

Enter the new admin page password (twice).

You can disable authentication by just pressing <Enter> (Blank for no password).

You can also remove the password by removing it from the configuration file (this requires the pihole-FTL service to be stopped, in order to save the changes).

```
sudo nano /etc/pihole/pihole.toml
```

Remove everything between the quotes (setting 'webserver.api.totp_secret').

```
pwhash = ""
```

9. Adding a local LAN list

If you installed lighttpd prior to installing pi-hole (see [here](#)), you can get to the pi-hole admin page by entering [http:// <Your Raspberry pi's IP address>:8080/admin](http://<Your Raspberry pi's IP address>:8080/admin). The remainder of this document assumes you will be using the http protocol on port 8080. In order to use <http://pi.hole:8080/admin> you need to make some configuration changes, see below.

This was achieved (older pi-hole versions) by creating a configuration file /etc/pihole/local.list. The IP addresses used to refer pi.hole can be configured, use your preferred [configuration](#) method.

Relevant settings:

```
[dns.reply.host]
force4 (default false)
IPv4 (default "")
force6 (default false)
IPv6 (default "")
```

The file /etc/pihole/local.list is now a static file (contains a warning only). The file is overwritten when running 'pihole -g ' or 'pihole -up'.

You can use the pi-hole solution to create custom names for your devices (different from the hostname), by using the web interface, click on 'Local DNS Records' and add the 'device name' and 'IP address'. Changes are saved in /etc/pihole/hosts/custom.list.

Due to the limitation of using the web interface, you can only use a single name for a device, example:

```
192.168.2.232 7730g
```

The [hosts file specification](#) however, allows multiple entries (names) for a single IP address, example:

```
192.168.2.232 7730geth0.internal 7730g.internal 7730geth0 7730g
```

Use the method below to overcome this limitation and / or **you plan to import your [clients into the database](#)**, using a script (sqlite3).

Both methods need a careful approach. In the section '[Raspberry pi installation](#)', '[Preparing your DHCP server](#)', I advised you to configure the DHCP server to make a static entry for the Raspberry pi (IP address – MAC address). The same approach should be used for clients. If the client isn't configured in such a way, it may receive a different IP address, when the DHCP lease has expired (read [this article](#), search for lease). The lease time usually (most routers) default to 14 days (this may vary, depending on make and model). As long as you connect to the network within that period, no IP change will occur, however, if you take a three week holiday, you might be in for a surprise.

Create /etc/pihole/hosts/localdns.list:

```
sudo nano /etc/pihole/hosts/localdns.list
```

Content, you need to replace the IP addresses, hostnames and domainname, to match your own environment, on Windows you can find your domain name by entering 'ipconfig /all' (look for Connection-specific DNS Suffix), on linux you can find your domain name by entering 'dnsdomainname'. Example:

192.168.2.5	s9.internal	s9
192.168.2.53	macbook.internal	macbook
192.168.2.102	ps4.internal	ps4

You can enter as many entries (lines) as you need.

The newly created list (/etc/pihole/hosts/localdns.list) is automatically used by pihole-FTL, as soon as it is changed. This is due to the dnsmasq setting "hostsdir=/etc/pihole/hosts" in /etc/pihole/dnsmasq.conf (This cannot be changed – hardcoded).

As soon as you've changed the file, you can use the dnsname to get to your devices, e.g., 'ping ps4', 'nslookup ps4', 'dig ps4'. Depending on your systems settings, it might be required to use the full name, e.g. 'ping ps4.internal'. You can change that, but these changes need to be done on the workstation (windows 10, linux, mac).

Warning: Not all the files in this manual are saved when using 'Teleporter' (web interface / Settings / Teleporter / Export). When customizing pi-hole, always keep a backup copy on a different machine.

10. Windows Whitelist.

The windows registry

(\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NlaSvc\Parameters\Internet) key, contains information, needed to determine the systems internet connection status.

In order to correctly update the windows internet status (network icon in the system tray) you need to add whitelist exceptions.

- Open the pi-hole admin page: <http://<Your Raspberry pi's IP address>:8080/admin/>
- Select "Domains" from the menu.
- Select the "Domain" tab, add the following entries (windows 7 and 10 entries are listed), select "Add to Whitelist":

```
www.msftncsi.com
dns.msftncsi.com
ipv6.msftncsi.com
www.msftconnecttest.com
ipv6.msftconnecttest.com
```

Pi-hole entries can be made, using the pi-hole admin page. The next section(s) will explain alternate methods to enter new entries.

An alternative approach would be to use a 'Regex filter' or 'Wildcard whitelist' entries. Remember that **whitelist entries always win**, if a domain matches a whitelist entry ('Allow list', 'Exact whitelist', 'Regex whitelist' or 'Wildcard whitelist'), pi-hole will allow the domain, regardless of what may be on the block lists.

It is recommended to be extremely careful, when using 'Regex whitelist' or 'Wildcard whitelist', because it might open the door for domains you don't want.

Example, the regular expression ("Regex filter" tab)

```
(\.|^)\.msftncsi\.com$
```

would be a replacement for 3 of the "Domain" entries in the above list, however, the domain tracker.msftncsi.com (example, doesn't currently exist) would also be allowed, which isn't really what we try to achieve. If you want to test what a new regular expression does or doesn't achieve, [regex101](#) allows you to test your regex, [regexper](#) explains how your regex works.

11. Modify lists using the 'pihole' command.

If you simply enter 'pihole' on the command line of your putty session, you will be presented with a list of possible options.

You can add an entry to a list by using the appropriate 'pihole' command.

Example: add the domain 'ipinfo.io' to the whitelist (always allow access to this site, even if the domain is in a block list)

```
pihole -w ipinfo.io
```

This method can be used to add whitelist (-w), blacklist (-b), wildcard blacklist domains (wildcard), regex blacklist domains (regex).

pihole-FTL will pick up the changes immediately, no need to restart pihole-FTL (part of the 'pihole' command).

12. Modify lists using sqlite3.

The pi-hole lists are no longer separate files (adlists.list, black.list, whitelist.txt), all data is now maintained in the sqlite3 database (/etc/pihole/gravity.db).

You can explore or modify the database by installing phpLiteAdmin, a web interface that will give you access to all sqlite3 databases. The installation (NOT part of pi-hole – NOT supported) is described [here](#).

You can add an entry to a list by using the appropriate sqlite3 command. The last version of pi-hole doesn't install the [sqlite3](#) package, sqlite3 is embedded in pihole-FTL.

Example: add the domain 'ipinfo.io' to the whitelist (always allow access to this site, even if the domain is in a block list).

```
sudo pihole-FTL sqlite3 /etc/pihole/gravity.db
insert or ignore into domainlist (domain, enabled) values ("ipinfo.io", 1);
.quit
```

OR add the domain and comment

```
sudo pihole-FTL sqlite3 /etc/pihole/gravity.db
insert or ignore into domainlist (domain, comment, enabled)
values ("ipinfo.io", "IP Address Data", 1);
.quit
```

Note you can split the command on multiple lines, ";" marks the end of the command. You will see "...>" on the screen after the first line, enter the second line after this.

pihole-FTL does NOT pick up the changes immediately, you'll need to reload the configuration.

```
pihole restartdns reload
```

Adding a single entry, using sqlite3 requires more effort than using the 'pihole' command, however, using sqlite3 is very useful to add multiple entries from a text file. This will be explained in the [regex](#) section.

13. Adding Wildcard sites to the blacklist.

Wildcard configuration (/etc/dnsmasq.d/03-pihole-wildcard.conf) is no longer supported by pi-hole, instead [regular expressions](#) are used.

You may still want to use wildcards, as it is a valid dnsmasq feature. FTLDNS, used by pi-hole, is based on dnsmasq. If you want to block an entire an entire domain and don't want to use regular expressions, create an additional configuration file for dnsmasq.

Warning! The entries in this list are **NOT affected** by the **disable** function in the pi-hole web interface. A change of the configuration file requires a restart ('sudo service pihole-FTL stop / sudo service pihole-FTL start')

Warning! The entries in this list do NOT trigger the [CNAME feature](#). If you want to create an entry that blocks CNAMEs, you need to use a regular expression.

```
sudo nano /etc/dnsmasq.d/wildcard.conf
```

In this example, we will block the entire ligatus.com domain, using null blocking. Add the following line to the file (only use a single entry per domain, either the IPv4 example **OR** the IPv4 and IPv6 example):

```
# Entry for IPv4 only
address=/ligatus.com/0.0.0.0

# Entry for IPV4 and IPv6
address=/ligatus.com/#
```

You can add multiple 'address' lines

Reload and restart the FTLDNS service

```
sudo service pihole-FTL stop
sudo service pihole-FTL start
```

14. Regular expressions.

As of pi-hole v4.0, regular expressions are used, to block domains. You can add wildcards or regular expressions, using the web interface (settings), the pihole command, or by editing the database. In pi-hole v4.1, PRIVACYLEVELS are introduced. By default, the level is 0 (Show everything). Regular expressions cannot be used if privacy level 4 (Disabled statistics) is used. The privacy level can be [changed](#) (Settings / Privacy / Query Anonymization).

To add a regular expression, using the 'pihole' command (example block 'doubleclick', notice the double quotes!):

```
pihole regex "(^|\.)doubleclick\.net$"
```

To add a regular expression using sqlite3 (example block 'facebook', notice the double quotes! - the regex can also be copied [here](#), discussion [here](#)):

```
sudo pihole-FTL sqlite3 /etc/pihole/gravity.db
insert or ignore into domainlist (domain, type, enabled)
values ('^(.+\.)?(facebook|fb(cdn|sbx)?|tfnw)\.[^.]+\$', 3, 1);
.quit
```

Note you can split the command on multiple lines, ";" marks the end of the command. You will see "...>" on the screen after the first line, enter the second line after this.

As you can see, as opposed to the sqlite3 [whitelist](#) example, a type field is used. It isn't used in the whitelist example, because the default value is zero. [Here](#), you can find an overview of the different types and some additional information on the domainlist database table.

Another example of regular expressions (block amp pages), reference [here](#):

```
^(.+[_.-])?amp(project)?\.
```

You can learn more about regular expressions in [this](#) pi-hole document. Examples of regular expressions are provided [here](#).

Adding a single entry, using sqlite3 requires more effort than using the 'pihole' command, however, using sqlite3 is very useful to add multiple entries from a text file. A well known and used list of regular expressions can be found [here](#). You can add these regular expressions by running /home/pi/regex.sh:



Click on the icon to view the script in a browser. Copy the content and make the script executable (sudo chmod +x /home/pi/regex.sh). You can also use [wget](#) (click on Basic usage) to copy the script directly. Execute the script with 'sudo'!

The web interface identifies a domain, blocked due to a regex match as follows (example):

AAAA	www.facebook.com	Blocked (regex blacklist)
------	------------------	---------------------------

Clicking on the link ('Blocked (regex blacklist)') will open a new page (<CTRL> click if your browser doesn't support this), showing the regex (highlighted), responsible for blocking.

<pre>^(.+.)??(facebook .*fb.+).(com net)\$</pre>	Regex blacklist ▾	Enabled <input checked="" type="checkbox"/>
--------------------------------------------------	-------------------	---------------------------------------------

You can either delete the regex, NOT recommended, as it will allow all domains, previously blocked due to the regex OR whitelist the domain on the first screen (query Log).

15. Deep CNAME inspection

Several articles (search for 'CNAME cloaking') describe the latest method advertisers and trackers use to bypass ad blockers. Naturally, ad blockers, including pi-hole, have found a solution for this.

'Deep CNAME inspection' is enabled by default, you can disable it (**NOT recommended**) by [changing](#) the setting 'dns.CNAMEdeepinspect'.

```
CNAMEdeepInspect = false
```

The web interface identifies a domain, blocked due to a CNAME match as follows (example):

AAAA	fonts.gstatic.com (blocked gstaticadssl.l.google.com)	Blocked (gravity, CNAME)
------	----------------------------------------------------------	--------------------------

This example shows fonts.gstatic.com is NOT in any block list (use 'pihole -q' to verify), but gstaticadssl.l.google.com (the CNAME) is (gstaticadssl.l.google.com is on 4 of my block lists), thus fonts.gstatic.com (the original request) will be blocked because of the CNAME. You can use the command 'dig @8.8.8.8 A fonts.gstatic.com' to verify the CNAME relation. This example is provided, using my block list set, it may not produce the same result on your system, because you don't use different block lists.

Pi-hole shows blocking, due to deep CNAME inspection, in the pi-hole log (/var/log/pihole/pihole.log). If a domain is blocked, due to deep CNAME inspection, you'll see something like (example used above):

reply fonts.gstatic.com is <CNAME>

reply gstaticadssl.l.google.com is blocked during CNAME inspection

Never whitelist a CNAME (gstaticadssl.l.google.com in the example). Several other, possibly undesired domains may use the same CNAME! In the above example, you would whitelist fonts.gstatic.com (**pi-hole developer recommendation**).

NextDNS, a DNS provider, has a [github page](#), explaining CNAME cloaking, and a list of companies (domains) that use this technique. In order to defend yourself against this practice, you can add these domains as regex (regular expressions), manually, or by running /home/pi/NextDNS.sh:



Click on the icon to view the script in a browser. Copy the content and make the script executable (sudo chmod +x /home/pi/NextDNS.sh). You can also use [wget](#) (click on Basic usage) to copy the script directly. Execute the script with 'sudo'!

The Adguard Team also has a [github page](#) (reported by [Sudoku](#)) with a list of companies (domains) that use this technique. Although a lot of the domains are already in the NextDNS list, you can add some additional domains (extracted from the [json file](#)) as regex (regular expressions), manually, or by running /home/pi/AdguardTeam.sh:



Click on the icon to view the script in a browser. Copy the content and make the script executable (sudo chmod +x /home/pi/AdguardTeam.sh). You can also use [wget](#) (click on Basic usage) to copy the script directly. Execute the script with 'sudo'!

16. Adding host lists.

1. Block Lists.

Pi-hole comes with a single blocklist (URL), used to create the gravity list. Pi-hole stores the URLs in the database (/etc/pihole/gravity.db / table adlist).

These URLs are used every Sunday, using a cron job, to update the gravity (domains) database (/etc/pihole/gravity.db / table gravity).

You can add entries to this table, using the web interface (Select "Lists" from the menu), whenever you add an entry to the list, using the web interface, you need to rebuild the gravity list, using "pihole -g" (terminal) or "Tools" / "Update Gravity" (web interface menu).

To add entries to the list manually, using sqlite3 (example):

```
sudo pihole-FTL sqlite3 /etc/pihole/gravity.db  
insert or ignore into adlist (address, enabled, type)  
values ('http://someonewhocares.org/hosts/', 1, 0);  
.quit
```

Note you can split the command on multiple lines, “;” marks the end of the command. You will see “...>” on the screen after the first line, enter the second line after this.

You can also add an entry with a comment, see [modifying lists using sqlite3](#).

Some URL's, containing lists I added:

```
http://someonewhocares.org/hosts/  
https://www.malwaredomainlist.com/hostslist/hosts.txt  
http://winhelp2002.mvps.org/hosts.txt  
http://www.hosts-file.net/download/hosts.txt  
http://v.firebog.net/hosts/Easyprivacy.txt  
# cryptojacking  
https://raw.githubusercontent.com/hoshisadiq/adblock-nocoin-list/master/hosts.txt  
https://gitlab.com/ZeroDot1/CoinBlockerLists/raw/master/list.txt
```

You should always check the format of a new host list, before adding it to your list. Not all lists can be parsed correctly.

An interesting set of lists, can be found [here](#). This page will let you choose from 3 sets of lists, I've been using the [non-crossed lists](#). This page will display a set of URLs that can be added. Using sqlite3, it is possible to add these list with a script. You can add these lists by running `/home/pi/firebog.sh`:



Click on the icon to view the script in a browser. Copy the content and make the script executable (`sudo chmod +x /home/pi/firebog.sh`). You can also use [wget](#) (click on Basic usage) to copy the script directly. Execute the script with 'sudo'!

Some lists aren't compatible with pihole and need to be pre-processed in order to use them. I have been using lists from this [site](#). Use the script to download and process the lists to your local file system.



Click on the icon to view the script in a browser. Copy the content and make the script executable (`sudo chmod +x /home/pi/quidsup.sh`). You can also use [wget](#) (click on Basic usage) to copy the script directly. Execute the script with 'sudo'!

After running this script, the result is a new folder `/home/pi/quidsup`. This folder contains two lists that can be used with pi-hole:

```
/home/pi/quidsup/notrack-blocklist.txt
```

```
/home/pi/quidsup/notrack-malware.txt
```

Note that gravity (pi-hole v6) is executed with user pihole, see [here](#), you'll need to allow the user pihole to read from the /home/pi/quidsup folder. In order to achieve this, execute the following commands:

```
# change the group of the folder
sudo chgrp pihole /home/pi/quidsup
# user pihole needs +x on /home/pi to read from /home/pi/quidsup
sudo chgrp pihole /home/pi
sudo chmod 710 /home/pi
```

Look at [this](#) topic for an alternative solution, if you don't want to change the access rights to /home/pi. You'll need to modify the script to change the target of the resulting lists.

To add these lists to your set of lists, using sqlite3 (or the web interface)

```
sudo pihole-FTL sqlite3 /etc/pihole/gravity.db
insert or ignore into adlist (address, enabled, type)
values ('file:///home/pi/quidsup/notrack-blocklist.txt', 1, 0);
insert or ignore into adlist (address, enabled, type)
values ('file:///home/pi/quidsup/notrack-malware.txt', 1, 0);
.quit
```

Notice we are using a local file, hence the "file:///".

If you have added new lists, using sqlite3 or a script, remember to activate the new lists:

```
pihole -g
```

Always remember to flush or [configure the DNS cache](#) on your workstation, to ensure correct responses, after adding new lists.

```
ipconfig /flushdns
```

Pi-hole v6 has the option to add '[allow lists](#)', lists that contain domains that will be allowed, despite one or more entries in blocklists. Be very carefull adding such lists, maintainers can be tricked into adding domains that really should be blocked, this may compromise your privacy.

2. Adblock Plus domain lists.

Pi-hole now provides limited support for Adblock Plus lists.

Adblock Plus lists are text files, the first line contains:

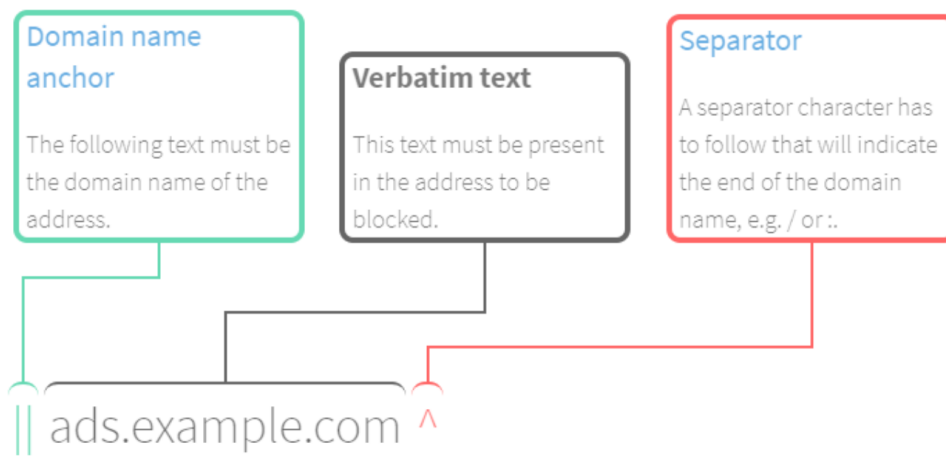
[Adblock Plus]

Regular Linux text files usually use the Number Sign (# - also referred to as the pound, hash or octothorpe sign). Adblock Plus text files use the Exclamation Mark (! also called the exclamation point).

If a list contains Adblock Plus entries, this will be logged in the gravity log (/var/log/pihole/pihole_updateGravity.log):

[!] List contained Adblock Plus style domains

Supported format is limited to (reference [here](#)) blocking by domain name (example 2)



This rule blocks:

- <http://ads.example.com/foo.gif>
- <http://server1.ads.example.com/foo.gif>
- <https://ads.example.com:8000/>

This rule doesn't block:

- <http://ads.example.com.ua/foo.gif>
- <http://example.com/redirect/http://ads.example.com/>

3. Subscribed allow lists (antigravity)

Pi-hole v6 allows users to add lists, containing domains that will be whitelisted. The methods, used to add allow lists are identical to [adding host lists](#) (block lists), the url is stored in the same database table (adlist), the type (new field in pi-hole v6) is 0 for block lists, 1 for allow lists. The whitelisted domains are stored in the antigravity table.

Both 'block lists' and 'allow lists' can be assigned to groups, using [Group Management](#), which will allow you to assign a specific list to a specific number of users, e.g. blocked for some users and allowed for others.

Remember that **whitelist entries always win**, if a domain matches a whitelist entry ('Allow list', 'Exact whitelist', 'Regex whitelist' or 'Wildcard whitelist'), pi-hole will allow the domain, regardless of what may be on the block lists.

It is recommended to be extremely careful, when using 'Allow Lists' (antigravity), because it might open the door for domains you don't want.

17. Group Management

Up to pi-hole v4.3.2, all block lists were applied to all clients. This often was a dilemma for the administrator, a specific user may need or request access to a specific web page, the only option to grant the request was to whitelist the domain, unfortunately, that whitelist entry affected all users.

Pi-hole uses group management to overcome this limitation, use this wisely!

The developer's documentation, contains very detailed examples, on how to use group management, and can be found [here](#).

It's NOT always easy to get the desired result. You could for example exclude an 'adlist' for a specific group, but the domains in that 'adlist' may exist in multiple lists. Always use:

```
pihole -q <domain>
```

This, to verify if a domain exists on multiple 'adlists'. Excluding a specific 'adlist' from a group is pointless, if the domains in the list exist in multiple 'adlists'.

Remember, 'whitelist' entries always have precedence. You can block what you want, if there is a 'whitelist' or 'regex whitelist' entry, the domain will never be blocked!

Changes made in Group Management don't appear to take effect until you restart pihole-FTL.

```
pihole restartdns reload-lists
```

1. Groups.

A default group exists, the 'The default group'. By default, all clients, even the ones not defined in the 'Clients' section, are member of this group, you can change membership in the 'Clients' section. You can add a new group, using the web interface (web interface menu / Groups)

2. Clients

By default, there are no configured clients. You can add a client, using the web interface (web interface menu / Clients). A client must be a member of at least one group, by default, this will be the 'The default group'. If a client isn't a member of any group, nothing will be resolved (no internet). If you are using both IPv4 and IPv6 on your network, and want to allow the client to bypass a specific 'Domain' or not use a specific 'Adlist', you need to add a 'Client' entry for both the IPv4 and IPv6 'Client' address.

This setup needs a careful approach. In the section '[Raspberry pi installation](#)', '[Preparing your DHCP server](#)', I advised you to configure the DHCP server to make a static entry for the Raspberry pi (IP address – MAC address). The same approach should be used for clients. If the client isn't configured in such a way, it may receive a different IP address, when the DHCP lease has expired (read [this article](#), search for lease). The lease time usually (most routers) default to 14 days (this may vary, depending on make and model). As long as you connect to the network within that period, no IP change will occur, however, if you take a three week holiday, you might be in for a surprise.

If you have created /etc/pihole/hosts/localedns.list, as described [here](#) (local LAN list), you can use a script to add the IP addresses and a comment (last entry of the line) to your pi-hole configuration.



Click on the icon to view the script in a browser. Copy the content and make the script executable (sudo chmod +x /home/pi/clients.sh). You can also use [wget](#) (click on Basic usage) to copy the script directly. Execute the script with 'sudo'!

3. Domains.

The 'Domains' list contains all whitelist, blacklist, regex whitelist and regex blacklist entries you have defined on your system. The steps, required to assign a whitelist, blacklist, regex whitelist or regex blacklist entry are very well explained in the [developer's documentation](#), thus NOT explained here.

4. Adlists

The 'Adlists' list contains all adlists you have defined on your system. The steps required to assign an 'adlist' entry are very well explained in the [developer's documentation](#), thus NOT explained here.

18. Group Management, a whitelist example.

I had a problem with my android devices, notifications didn't work anymore. I found a [hint](#) to the solution. Turns out a number of block lists started adding mtalk.google.com entries (example: alt2-mtalk.google.com).

A solution, using a **regex whitelist** entry, for **android devices only**:

1. Create the whitelist regex entry (web interface menu / tab "Regex filter" / Add to Whitelist):

```
^((alt)[0-9](-))?mtalk\.google\.com$
```

Result will look like this:

<code>^((alt)[0-9](-))?mtalk\.google\.com\$</code>	Regex whitelist ▾	Enabled	mtalk.google.com (andr
----------------------------------------------------	-------------------	---------	------------------------

2. Create a new group 'android' (web interface menu / Groups)

android	Enabled	android devices
---------	---------	-----------------

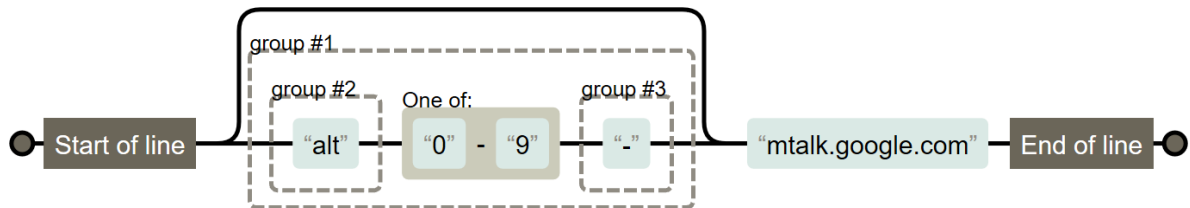
3. Add the android device (web interface menu / Clients), you've already imported (see the [clients](#) script) to the 'android' group. Ensure the devices are assigned to all the groups, e.g. if the 'android' group was the first you created, the entry will look like this (IP address and client name will differ), repeat this for all android devices:

192.168.2.170	s9	All selected (2) ▾
---------------	----	--------------------

- Assign the domain entry to the android group. Ensure the domain (whitelist regex) is only assigned to the 'android' group (web interface menu / Domains).

<code>^((alt)[0-9](-))mtalk\.google\.com\$</code>	Regex whitelist ▾	Enabled	mtalk.google.com (andr	android ▾
---------------------------------------------------	-------------------	---------	------------------------	-----------

All android devices will now receive a valid response, when requesting info for the mtalk.google.com domain and specific subdomains. The regex accomplishes the following (analyze a regex [here](#)):



Another example of selective whitelisting (whitelisted google ad links from the google search results) can be found in [this](#) document.

19. Group Management, duplicate blacklist and whitelist entries

Pi-hole allows you to create an identical blacklist AND whitelist entry.

An example: Assuming you want to block facebook, but allow access for a specific device (or devices).

The regular expression:

```
^(.+\..)?(facebook|fb(cdn|sbx)?|tfbnw)\. (co\..)?[^.]+$
```

- Create a blacklist regular expression for facebook and assign it to the default group, this will block facebook for all clients:

Domain/RegEx	Type	Status	Comment	Group assignment
<code>^(.+\..)?(facebook fb(cdn sbx)? tfbnw)\. (co\..)?[^.]+\$</code>	Regex blacklist ▾	Enabled	block facebook	Default ▾

- Create a group to allow access to facebook:

Name	Status	Description
Default	Enabled	The default group
AllowFacebook	Enabled	Allow Facebook

3. Add a client(s) to the group, assign the client to both the default and the AllowFacebook groups:

IP address	Comment	Group assignment
192.168.2.170 s9.localdomain	s9	<div>All selected (2) ▾</div> <div>Apply</div> <div>All None</div> <div>Default ✓</div> <div>AllowFacebook ✓</div>

4. Create the whitelist regular expression (identical to the blacklist regular expression) for facebook and assign it to the AllowFacebook group:

Domain/RegEx	Type	Status	Comment	Group assignment	Action
<code>^(.+\.?)?(facebook fb(cdn sbx)? tfbnw)\.(co\.)?[\.\-]+\$</code>	Regex whitelist	Enabled	allow facebook	AllowFacebook	
				<div>Apply</div> <div>All None</div> <div>Default</div> <div>AllowFacebook </div>	

Done, facebook will now be accessible for the specific device(s) only, this because a **whitelist entry always wins**.

20. Windows DNS cache.

Enable/Disable pi-hole, using the pi-hole admin console, will not have an effect unless you change the windows DNS cache time permanently

To disable the Windows DNS cache:

Create a registry file with the following contents and add the info to the registry:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\Dnscache\Parameters]
"MaxCacheTtl"=dword:00000001
```

Double click the file to add the setting to the registry.

To enable the Windows DNS cache:

Create a registry file with the following contents and add the info to the registry:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\Dnscache\Parameters]
"MaxCacheTtl"=-
```


Double click the file to add the setting to the registry.

21. Changing the hostname.

If you have multiple pi's running, you might want to change the hostname, this to easily identify the host you're working on, using putty. Enter the following command:

```
sudo hostnamectl set-hostname <newhostname>
```

If you've installed [webmin](#), you may need to change an additional setting in the webmin configuration.

- Webmin / Webmin configuration / Sending Email:
 - o Requires [mail setup](#)!
 - o From address for email from Webmin:
 - Select **Address**
 - Enter the (g)mail address you used in the mail setup.

22. Protect your Raspberry Pi.

We've already enabled [key authentication](#), changed the [UNIX password](#) and [disabled password logon](#), we can however increase the security even more.

Depending upon you paranoia level, you can apply all security measures, described [here](#), however this document is limited to MITM attacks, spoof protection and disabling routing.

```
sudo nano /etc/sysctl.conf
```

Remove the comment sign from the lines below (red comment signs only)

```
# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# Additional settings - these settings can improve the network
# security of the host and prevent against some network attacks
# including spoofing attacks and man in the middle attacks through
# redirection. Some network environments, however, require that these
# settings are disabled so review and enable them as needed.
#
# Do not accept ICMP redirects (prevent MITM attacks)
#net.ipv4.conf.all.accept_redirects = 0
#net.ipv6.conf.all.accept_redirects = 0

# Do not send ICMP redirects (we are not a router)
#net.ipv4.conf.all.send_redirects = 0

# Do not accept IP source route packets (we are not a router)
#net.ipv4.conf.all.accept_source_route = 0
#net.ipv6.conf.all.accept_source_route = 0
```

Reboot the Raspberry pi

```
sudo reboot
```

23. Disable unused hardware (Raspberry Pi® 3B, 3B+, 3A+, 4B and Zero W).

You may want to disable the WIFI interface and Bluetooth (reference [here](#)).

1. Disable WIFI.

```
echo "dtoverlay=disable-wifi" | sudo tee -a /boot/config.txt
```

2. Disable Bluetooth.

```
echo "dtoverlay=disable-bt" | sudo tee -a /boot/config.txt  
sudo systemctl disable hciuart
```

Reboot the Raspberry pi

```
sudo reboot
```

24. Using cron.

[cron](#) is installed by default on [Raspbian](#), no need to install additional packages. [Webmin](#), if installed, lists all cron jobs (system / scheduled cron jobs). Although many users prefer to use crontab to add cron jobs, jobs specified in /etc/cron.d are also processed, see the [man page](#) for details. I have been using this directory to add cron jobs, pi-hole development also uses this method (the pi-hole job - /etc/cron.d/pihole).

Some scripts in the github repository need to be executed regularly. Some examples:

1. Add new [NextDNS](#) entries (weekly).
Add the following file: /etc/cron.d/NextDNS (assuming your script is /home/pi/NextDNS.sh):

```
10 23 * * 6 root PATH="$PATH:/home/pi/" /home/pi/NextDNS.sh >/dev/null 2>&1
```

This script needs to run before the weekly gravity update, scheduled on Sunday, the above settings will run the script on Saturday at 23:30.

2. Run [needrestart](#) (daily).
Add the following file: /etc/cron.d/needrestart (assuming your script is /home/pi/needrestart.sh):

```
30 5 * * * root PATH="$PATH:/home/pi/" /home/pi/needrestart.sh >/dev/null 2>&1
```

This script will be executed daily at 05:30, it will send a mail (mail setup required) if user intervention is required.

- The redirection (>/dev/null 2>&1) ensures there will be no undesired emails, indicating the cron job has been executed. In order to troubleshoot cron jobs, remove the redirection temporarily.

- Make sure the cron instruction is terminated with a line feed, the instruction will not be processed without it.

25. Install some useful system tools

1. Install watchdog

Your system might get into trouble, by running a CPU hogging script, or a package that misbehaves. This might lead to an inaccessible (unresponsive) system, leaving you no choice, but to cut the power to regain access, **which might cause damage to your SD card**. Raspbian has a kernel module (bcm2835_wdt) that can help you avoid this drastic intervention.

Enable watchdog to send [mails](#), whenever triggered:

```
sudo ln -s /usr/bin/msmtp /usr/lib/sendmail
```

Example mail (high temperature):



raspberrypi.localdomain is going down!

Message from watchdog:

It is too hot to keep on working. The system will be halted!

Syslog entries:

```
watchdog[924]: temperature increases above 76 (/sys/class/thermal/thermal_zone0/temp)
watchdog[924]: temperature increases above 78 (/sys/class/thermal/thermal_zone0/temp)
watchdog[924]: temperature increases above 78 (/sys/class/thermal/thermal_zone0/temp)
watchdog[924]: it is too hot inside (temperature = 80 >= 80 for /sys/class/thermal/thermal_zone0/temp)
watchdog[924]: shutting down the system because of error 252 = 'too hot'
```

Install watchdog:

```
sudo apt-get -y install watchdog
```

Configure the watchdog

```
sudo nano /etc/watchdog.conf
```

Uncomment the following lines, don't uncomment "max-load-1", this may cause unexpected reboots!

```
max-load-5          = 18
max-load-15         = 12
watchdog-device      = /dev/watchdog
```

Uncomment the following line and change the value (default 60, change to 15):

```
watchdog-timeout = 15
```

Reboot to start the watchdog with these settings:

```
sudo reboot
```

The watchdog will now be able to reboot your system, as soon as the load on the system is excessive. Normal load can be monitored on the pi-hole admin page (upper left corner) or by entering

```
uptime
```

2. Install schedtool

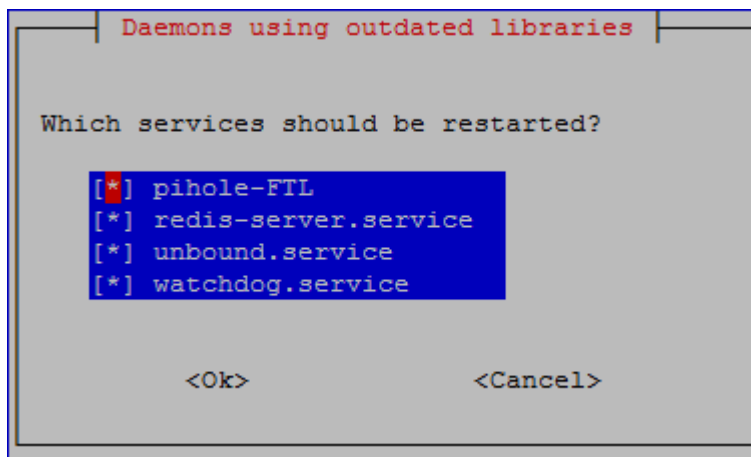
Some packages can use schedtool, to optimize the CPU use and cores. Install the package to allow CPU use optimization.

```
sudo apt-get -y install schedtool
```

3. Install needrestart.

If you installed and configured daily automatic updates, using [Webmin](#), your system will be kept up to date automatically, however Webmin doesn't restart daemons, still using the outdated libraries. The package needrestart allows you to determine which daemons need a restart. You will need to restart the daemon manually.

Needrestart integrates with apt-get, so running 'sudo apt-get -y upgrade' will automatically trigger the test. Webmin doesn't trigger needrestart, you will have to check it manually (sudo needrestart), or have a daily [cron job](#) and script to check it for you, and send you a mail (requires [mail setup](#)).



Installing needrestart:

```
sudo apt-get -y install needrestart
```

On a raspberry pi, some settings need to be modified in the configuration file, we don't want to test kernel updates and microcode updates, as these tests will generate false positives. Edit the configuration file:

```
sudo nano /etc/needrestart/needrestart.conf
```

Uncomment the following settings, change the values if necessary (value 0 recommended).

```
$nrconf{kernelhints} = 0;  
$nrconf{ucodehints} = 0;
```

These settings are commented out (#) in the original configuration file (near the end of the file), uncomment and change the value where required.

To avoid having to check things manually, the following script will perform the required checks, and send you an email. Create a [cron job](#) to execute it daily.



Click on the icon to view the script in a browser. Copy the content and make the script executable (sudo chmod +x /home/pi/needrestart.sh). You can also use [wget](#) (click on Basic usage) to copy the script directly. Execute the script with 'sudo'!

4. Helping the RANDOM number generator.

To avoid unexpected messages and warnings (you might see messages like 'System doesn't have enough entropy') whenever the system needs a random number, install rng-tools. As of the April 2019 version of Raspbian, this package is already installed, you only need to modify the configuration file.

```
sudo apt-get -y install rng-tools
```

Edit the configuration file

```
sudo nano /etc/default/rng-tools-debian
```

Add the following line (**bold** only):

```
#HRNGDEVICE=/dev/hwrng  
#HRNGDEVICE=/dev/null  
HRNGDEVICE=/dev/urandom
```

26. IPv6 address.

If you are using the default [blocking mode](#), the value of the IPv6 address, registered by the pi-hole installation doesn't really matter, unless you use the address to browse to the admin web interface. If however, you are using a blocking mode where the IPv6 address is used, you may be confronted with the problem your ISP hands out IPv6 addresses that changes regularly. To overcome this problem (implies you have completed [mail setup](#)):

Create a script (/home/pi/IPv6check.sh) with the following content (you need to update '<your account name>'):



Click on the icon to view the script in a browser. Copy the content and make the script executable (sudo chmod +x /home/pi/IPv6check.sh). You can also use

[wget](#) (click on Basic usage) to copy the script directly. Execute the script with 'sudo'!

Using GUA: Replace the first few digits ('2a02' in my case) in the grep command to match your own!!!

Using LUA: Replace '2a02' in the grep command with 'fc\|fd'

Make the script executable

```
sudo chmod +x /home/pi/IPv6check.sh
```

Create a [cron job](#) (/etc/cron.d/IPv6check) with the following content:

```
19 7 * * * root PATH="$PATH:/home/pi/" /home/pi/IPv6check.sh
```

Change the time to something appropriate for your system, the example runs at 07h19

27. Backup your pi-hole.

This procedure will only work, if you manually created a smaller Linux partition, explained in this [document](#). If you didn't reserve some free space, to allow for successful cloning, you probably will have to shrink the Linux partition first.

Once you have a working pi-hole, you can avoid setting it all up again by creating an image of your system.

Shutdown your system

```
sudo poweroff
```

Remove the SD card from the Raspberry Pi.

Optional: Use [DiskGenius](#) to check SD card integrity, see [here](#).

Use [Win32DiskImager](#) to create an image

- Insert the SD card into your computer.
- Start Win32DiskImager.
- Image File: Select a location and name for the image, e.g. C:\temp\pi-hole.img
- Device (source): Select the drive, holding the SD card, multiple partitions (drives) exist on the SD card, select the first one (all partitions will be processed).
- Select **Read**

Wait...

Use [Win32DiskImager](#) to write the image

- Insert the new (or used) SD card into your computer.
- Format the SD card, using [SDFormatter](#).
- Start Win32DiskImager.
- Image File: Select a location and name for the image, e.g. C:\temp\pi-hole.img
- Device (target): Select the drive, holding the SD card.

- Select **Write**

Wait...

Whenever you cloned (use [Win32DiskImager](#)) a backup image, the first thing you should do is set the date and time and restart the [NTP service](#).

If you have configured your system to use the built-in pihole-FTL NTP services, no actions are required. Pihole-FTL will apply the correct time automatically.

If you are using 'systemd-timesyncd.service', the systems default NTP service requires the following command to allow for time to be synchronized (use the correct date and time).

```
sudo timedatectl set-time "2024-07-20 09:34:00"
```

It may take a while, but eventually, time will be synchronized, you can verify this by entering:

```
timedatectl
```

Look for "System clock synchronized: yes".

If you are using 'chrony', the systems default NTP service requires the following command to allow for time to be synchronized. This will update your system clock quickly (might break some running applications – reboot if required).

```
sudo chronyc -a makestep
```

28. Customizing pi-hole (optional).

This section assumes you have installed a web server, as suggested, prior to [installing pi-hole](#). In order to ensure the php scripts will work, the following php packages needs to be installed (assuming lighttpd web server):

```
sudo apt-get -y install php-common
sudo apt-get -y install php-cgi
sudo apt-get -y install php-sqlite3
sudo apt-get -y install php-xml
sudo apt-get -y install php-intl
sudo apt-get -y install php-json
sudo apt-get -y install php-mbstring
sudo apt-get -y install php-dev
```

Execute the following to enable fastcgi (assuming lighttpd web server):

```
sudo lighttpd-enable-mod fastcgi-php
```

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

The scripts in this section assume you have a working, php enabled web server up and running.

1. PHP info.

Pi-hole relies on PHP. If you want to add your own PHP scripts, you need to know how PHP is configured.

Create a file info.php in /var/www/html, content:

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

To view the PHP info, browse to:

```
http://<your Raspberry pi's static address>/info.php
```

2. Lighttpd Server Status.

The default web server (lighttpd) has a status page, however, it isn't activated with a default pi-hole installation.

Reference: https://redmine.lighttpd.net/projects/1/wiki/Docs_ModStatus

To enable this page, run the following script (/home/pi/lighttpdstatus.sh):



Click on the icon to view the script in a browser. Copy the content and make the script executable (sudo chmod +x /home/pi/lighttpdstatus.sh). You can also use [wget](#) (click on Basic usage) to copy the script directly. Execute the script with 'sudo'!

Make the script executable:

```
sudo chmod +x /home/pi/lighttpdstatus.sh
```

Unfortunately, this configuration will be lost, as soon as you upgrade (pihole -up) or reconfigure (pihole -r) your system. Simply run the script again, the script will add the missing entries.

Run the script, assuming the current directory is /home/pi.

```
sudo ./lighttpdstatus.sh
```

To view the Server status, browse to:

```
http://<your Raspberry pi's static address>/server-status?refresh=15
```


3. Browsing the FTL database.

The FTL database is a [sqlite3](#) database, it contains the data, required to populate pi-hole's admin website.

Reference: <https://www.phpliteadmin.org/download/>

You can browse this database by installing phpLiteAdmin. To enable this, run the following script (/home/pi/phpliteadmin.sh):



Click on the icon to view the script in a browser. Copy the content and make the script executable (sudo chmod +x /home/pi/ phpliteadmin.sh). You can also use [wget](#) (click on Basic usage) to copy the script directly. Execute the script with 'sudo'!

If you encounter problems running phpliteadmin, you can switch to the development version of phpliteadmin (see note on the download page). Change the wget download into:

```
wget https://www.phpliteadmin.org/phpliteadmin-dev.zip
```

Make the script executable:

```
sudo chmod +x phpliteadmin.sh
```

Run the script, assuming the current directory is /home/pi

```
sudo ./phpliteadmin.sh
```

To access the database (password: '**admin**'), browse to:

```
http://<your Raspberry pi's static address>/phpliteadmin.php
```

Information regarding the content of the database can be found [here](#).

The database will be accessible in read-only mode, you cannot make any changes. To make the database writable, execute the following commands:

```
# Write permissions on the Database
sudo usermod -a -G pi-hole www-data
sudo chmod 775 /etc/pi-hole
sudo chmod 664 /etc/pi-hole/pi-hole-FTL.db
sudo chmod 664 /etc/pi-hole/gravity.db
sudo service lighttpd stop
sudo service lighttpd start
```

Useful chromium extension [here](#) (Epoch Convertor by drautb, converts Unix timestamp to date & time). Example:

← T →	id	timestamp	type	status	domain	client
	723150	1655624859	1	2	ds.download.windowsupdate.com	192.168.2.2
	723151	1655624859	1	2	ds.download.windowsupdate.com	192.168.2.2
		06/19/2022 - 09:47:39		1	userlocation.googleapis.com	192.168.2.1
				1	ssl.google-analytics.com	192.168.2.1
	723154	1655624901	1	2	fe2.update.microsoft.com	192.168.2.2
	723155	1655624901	1	2	fe2.update.microsoft.com	192.168.2.2

29. DoT / DoH

Pi-hole will only filter DNS requests, if your devices use pi-hole as their only DNS server. Some devices (example Chromecast), use a hard coded DNS server (example [8.8.8.8](#)), the DNS requests from these devices will never be processed by pi-hole. To overcome this, it is necessary to implement a firewall rule, that redirects all DNS requests ([port 53](#)), **not originating from pi-hole**, to pi-hole.

Pi-hole cannot be used to filter DoT ([DNS over TLS](#)) requests, the DoT request will have a DOT server as destination, thus bypassing pi-hole. Fortunately, it's easy to block DoT requests on your network, by implementing a firewall rule that blocks [port 853](#), thus effectively killing DoT.

Pi-hole cannot be used to filter DoH ([DNS over HTTPS](#)) requests, the DoH request will have a DoH server as destination, thus bypassing pi-hole. DoH cannot be easily blocked, because it uses [port 443](#), which happens to be the same port, used for HTTPS.

Unfortunately, some applications have implemented DoH capability, thus allowing the application to completely ignore the systems DNS settings. Those applications will thus bypass pi-hole, and have unfiltered access to the internet.

Pi-hole implemented a way to [prevent Firefox from using DoH](#), by ensuring the domain use-application-dns.net ([canary domain](#)) is resolved as an NXDOMAIN, thus informing Firefox to use the DNS system settings.

Pi-hole also implemented a way to warn the user he should use a specific network without using a [Private Relay](#) (icloud private Relay, the apple version of [oDoH](#)).

You can read more about this [here](#) (scroll down to section 6 – Network control, using DNS entries).

Other applications however, do not have a straight forward (or none at all) method, to ensure they don't use DoH. The only way to prevent these applications from using DoH, is to ensure the DoH destination server cannot be reached. There are a number of sites that list information, regarding existing DoH servers. Available to you now, compiled from several sources, updated daily, are two lists ([DOHipv4.txt](#) and [DOHipv6.txt](#)), you can use to implement a firewall rule, blocking port 443 for the destinations in the list. **You must ensure to block only port 443** for these IP's, the list contains the IP's for servers that reply to both port 443 and port 53, blocking all access to these IP's could possibly break pi-hole.

Unfortunately, some of the IP addresses in the above lists, cannot be blocked, as the IP is the destination for both DoH and content. For example, the DoH server dns.cloudflare.com has the same IP(s) as cdnjs.cloudflare.com, the later is used to serve some scripts, used by

several by several websites, such as linuxquestions.org. If the IP for dns.cloudflare.com (DoH) is blocked, the webpage will load extremely slow, as it cannot load the required content. To overcome this, two additional lists ([DOHexceptionsIPv4.txt](#) and [DOHexceptionsIPv6.txt](#)) are provided. The idea here is to create exceptions (allow) these addresses for **specific devices only**.

In order to use the list(s), you will need to create a Firewall [URL Table Alias](#) and a firewall rule that blocks port 443 for the DoH server destinations, using the alias. **This will NOT be possible on all firewalls**, check your firewalls documentation.

A method to setup the required aliases and rules on [pfsense](#), is described in this [document](#).

The [GitHub](#) repository provides the (o)DoH domain [list](#) in rpz format, which eliminates the need to generate a pi-hole compatible list from the [database](#), see the [unbound](#) section for details.

30. Using unbound as upstream resolver.

In order to increase DNS privacy, many pi-hole users use [unbound](#) as upstream resolver. Basic unbound installation instructions, as recommended by the pi-hole developers, can be found [here](#). Unbound has a lot of configuration options, most of them explained [here](#). Optimizing the performance of unbound can be achieved, using these [guidelines](#).

Unbound v1.10.0 allows the use of [response policy zones](#) (rpz), however, v1.14.0 has added a lot of configuration options, v1.14.1 does allow pi-hole to detect (query log) domains, blocked by the unbound rpz configuration. This will allow you to block domains without the option to whitelist, even when pi-hole is temporarily disabled. Unbound automatically downloads and updates these zones, using the settings in the SOA record (part of the [rpz file](#)). You can find more on this (how to setup) in this [document](#).

31. Change Log

18-02-2025

- Initial release of pi-hole version 6.

17-03-2025

- [Webmin](#) 2.303 released.

30-03-2025

- [Pi-hole core v6.0.6](#), [Pi-hole web v6.1](#), [Pi-hole-FTL v6.1](#) released.