Einstein's Theory of ad blocking

A guide to Pi-hole and Unbound

Alex "AK" Kras

Pi month @ the Undercroft | May 21st, 2020

Great poster Chris!

First off!

THE UNDERCROFT

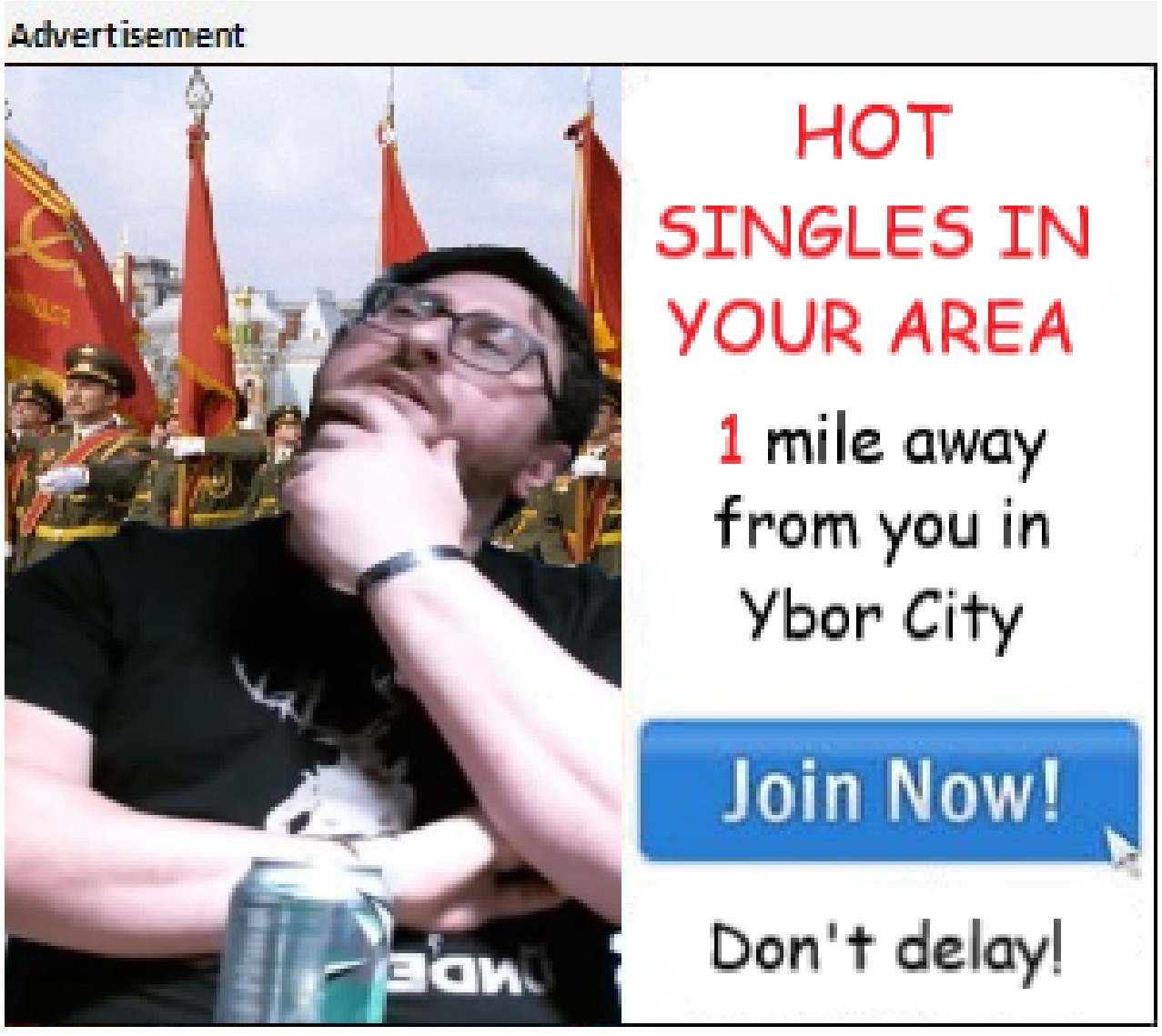
I want to give a big thanks to the Undercroft community for allowing me to give this talk.

- Huge props to Nestor for reviewing my slides and helping get stuff setup for the talk!
- Thanks to Chris, Jon, Adam and all the speakers who helped to keep us occupied during quarantine with the live-streams!

Thanks to Dan as well!

Huge shout out to Dan Schaper @ Pi-hole for sending over some stickers for you guys!

Who's sick of seeing stuff like this?



Don’t forget about malicious content!

Hackers have breached 60 ad servers to load their own malicious ads

Why buy legitimate ad slots to deliver malvertising when you can just hack the server instead.

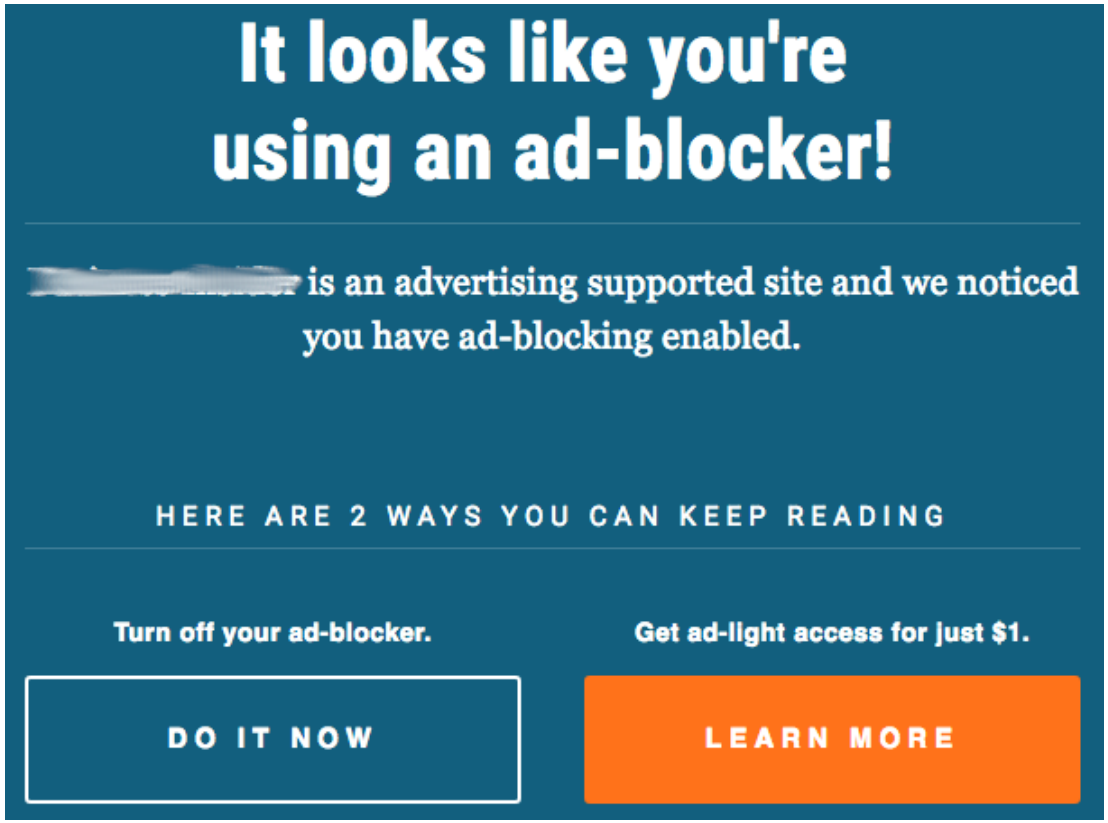
Major sites including New York Times and BBC hit by 'ransomware' malvertising

Google's Doubleclick ad servers exposed millions of computers to malware

Hackers Use Chipotle Ad To Spread Malware

How do we get rid of them?

The typical answer you'll hear is to download an "ad blocker". This usually refers to some sort of browser extension or add-on that blocks content/assets on the webpage. These are known to have their limitations.



Traditional ad-blocker limitations

These work fairly well inside a browser but what happens when you want to take it a step further. What if you wanted to block Windows 10 Telemetry or a chatty smart TV. Maybe you even want to make sure your kids aren't tempted to click ads in their mobile games. This is where the traditional "ad blocker" could use some help.

Some are also better than others

Ad Blockers Are Making Money Off Ads (And Tracking, Too)

Adblock Plus doesn't block *all* ads, but rather operates what it calls an "acceptable ads" program, where ads that meet its criteria for things like placement, size, and distinction, are "whitelisted"—that is, if the company displaying the ads is willing to split the revenue gained by whitelisting with Adblock Plus. Companies can apply to have their sites whitelisted, but Adblock Plus has also reached out to some to solicit their business. Other ad blockers, such as mobile app Crystal, take a similar whitelisting approach.

Google Saved An Estimated $887 Million By Paying Adblock Plus To Show Its Ads

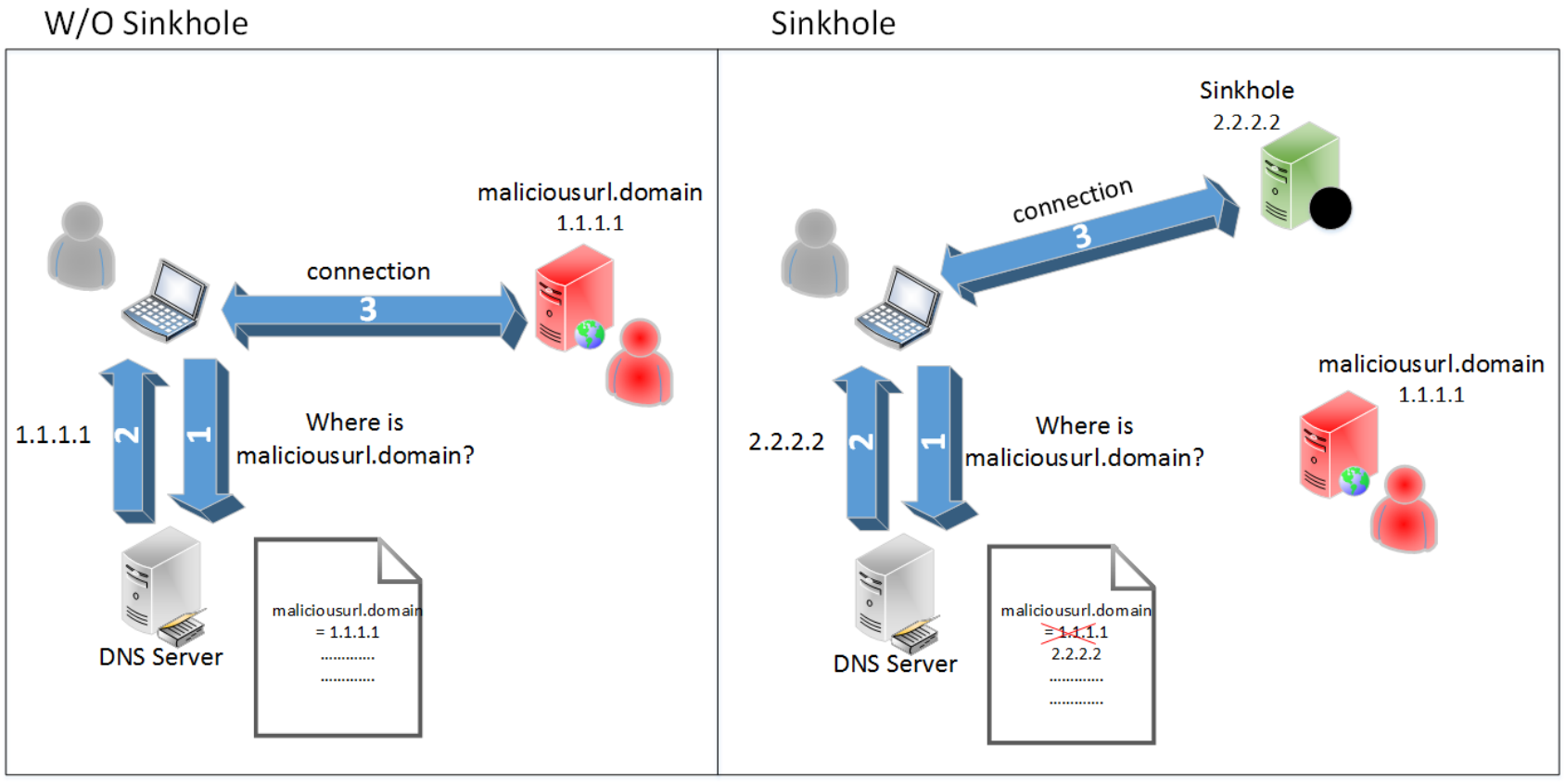
What is Pi-hole?

Simply put, it's a network-wide ad blocker that effectively works as a DNS sinkhole.

- Open-source (under EUPL).
- Lightweight, perfectly capable even on SBC's such as an RPi Zero.
- Easy to install, configure, and use.
- Perfect for devices like smartphones, Smart TV's, IoT, etc. which typically lack the ability to block ads on client-side.

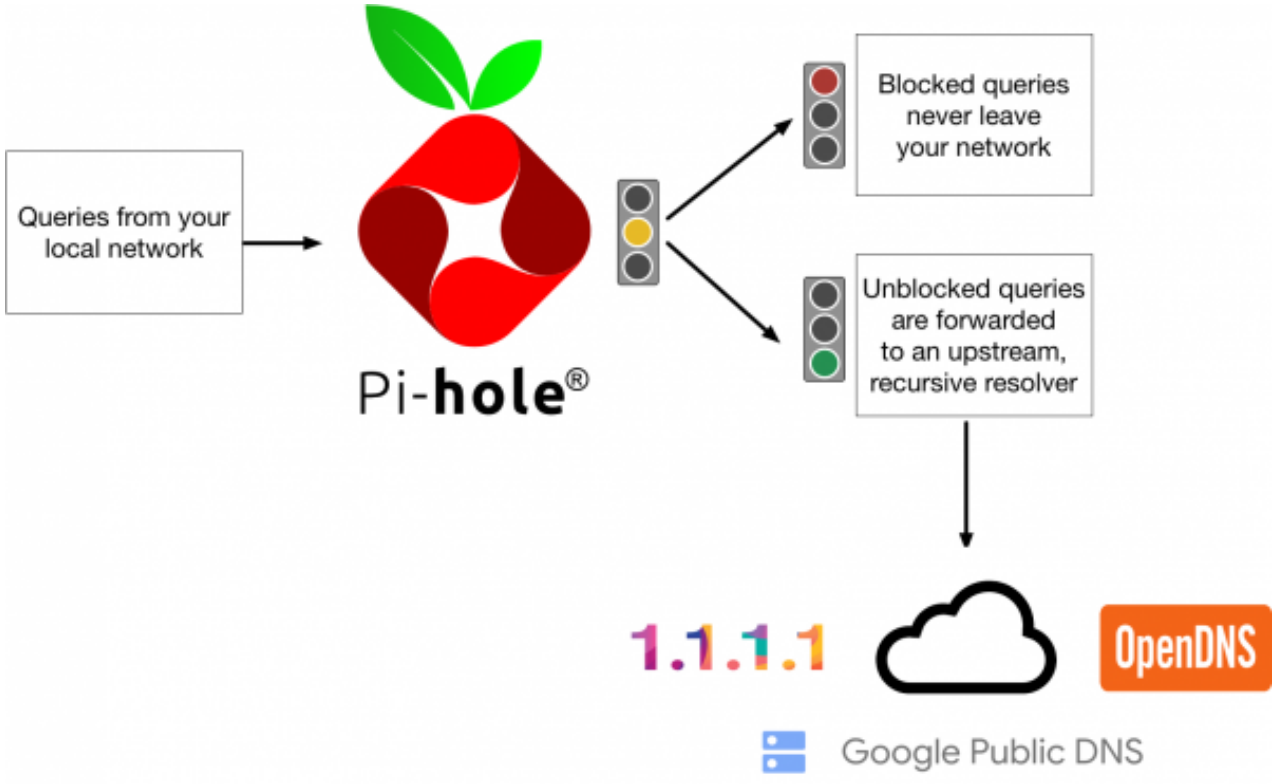
DNS Sinkhole

These work by having the DNS forwarder intercept outbound queries and return a “false” IP, typically non-routable, to the client when they attempt to connect to a domain from a specified blocklist.



Pi-hole blocking mode

The *default blocking mode* on Pi-hole, answers blocked queries with an unspecified address (0.0.0.0 or ::, for IPv4). It's important to understand that the Pi-hole handles this locally and then forwards unblocked queries to the upstream DNS provider of your choice.



Faster-than-light!

Historically, Pi-hole relied on Dnsmasq for the actual DNS capabilities, among PHP and others for interaction with the logs and web interface. In 2017, they debuted their faster-than-light engine.

- Forked Dnsmasq for DNS caching, forwarding and DHCP capabilities
- Interactive API
- Lightweight and efficient
- Long-term storage (SQLite3)

V5.0 is here!

Pi-hole V5.0 is here as of May 10th. Bringing with it a slew of new features.

- Group management
  - Per-client tracking
  - More granular control over clients use of adlists, etc.
- New Gravity DB
- Deep CNAME inspection
- A number of other smaller tweaks!

Minimum Requirements

The latest versions of Pi-hole and FTLDNS, make it extremely capable on hardware with limited resources. The RPi Zero is commonly used with Pi-hole, although note the lack of ethernet.

Minimum Requirments:

- 52MB of free space
- 512MB RAM

In addition, it's easily deployable inside a Virtual Machine or Docker container.

Supported Distros

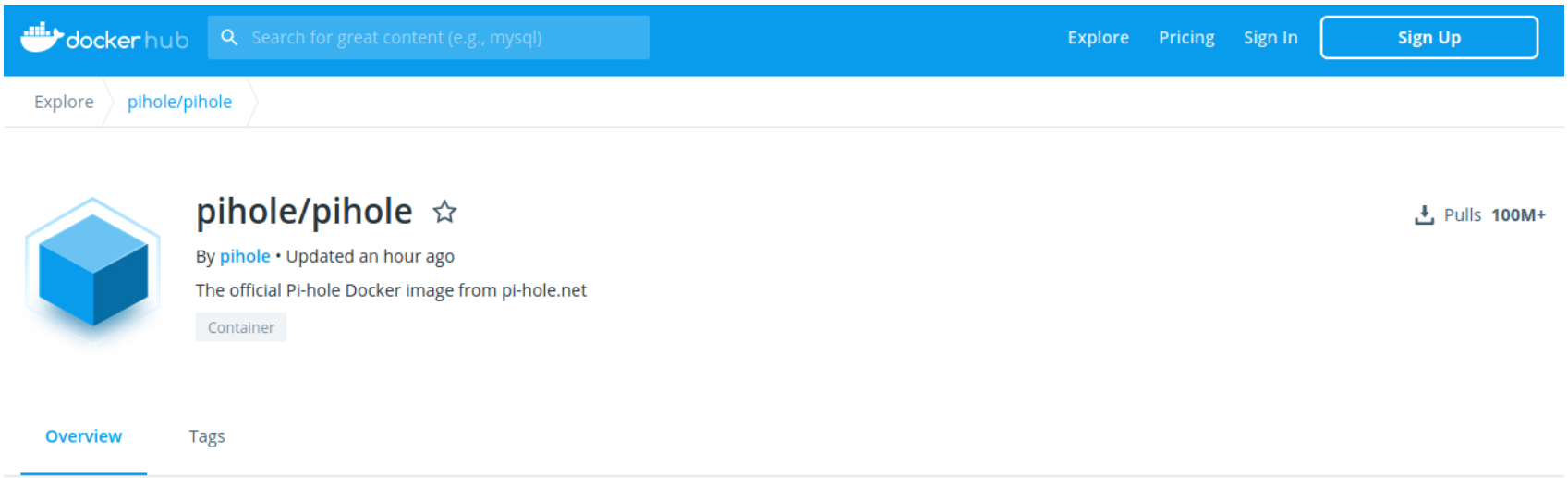
Pi-hole can run on most modern GNU/Linux distributions.

The following operating systems are **officially** supported:

Distribution	Release	Architecture
Raspbian	Stretch / Buster	ARM
Ubuntu	16.x / 18.x	ARM / x86_64
Debian	9 / 10	ARM / x86_64 / i386
Fedora	28 / 29	ARM / x86_64
CentOS	7	x86_64

Docker

For more info on the official Pi-hole image, head over to their *Github* or *DockerHub* for more info.



Ports

The typical install with the web server will use ports 80 (HTTP) and 53 (DNS).

Service	Port	Protocol	Notes
● dnsmasq	53 (DNS)	TCP/UDP	If you happen to have another DNS server running, such as BIND, you will need to turn it off in order for Pi-hole to respond to DNS queries.
dnsmasq	67 (DHCP)	IPv4 UDP	The DHCP server is an optional feature that requires additional ports.
dnsmasq	547 (DHCPv6)	IPv6 UDP	The DHCP server is an optional feature that requires additional ports.
● lighttpd	80 (HTTP)	TCP	If you have another Web server already running, such as Apache, Pi-hole's Web server will not work. You can either disable the other Web server or change the port on which `lighttpd` listens, which allows you keep both Web servers running.
pihole-FTL	4711-4720	TCP	FTL is our API engine and uses port 4711 on the localhost interface. This port should not be accessible from any other interface.

Hardware

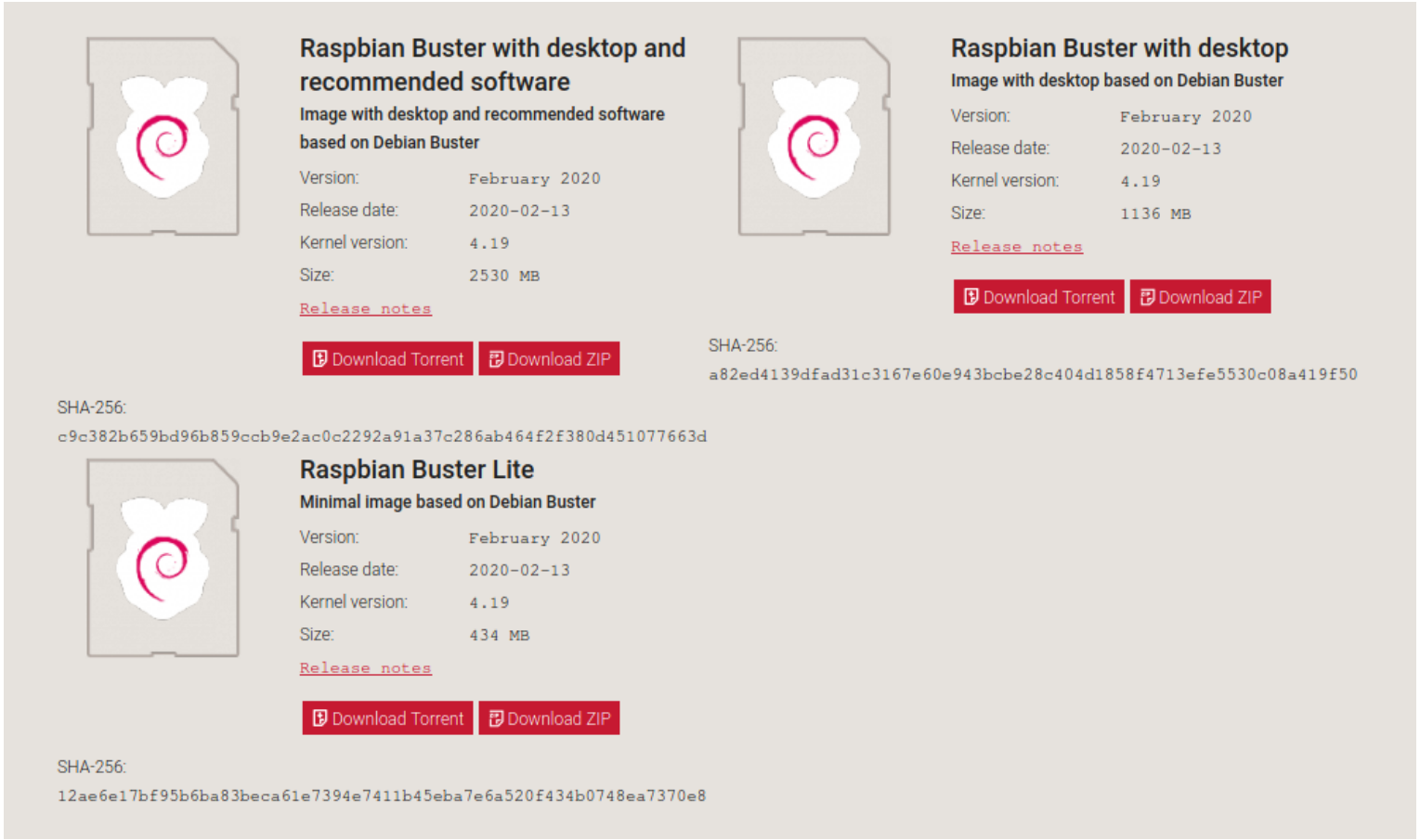
Since it is Pi month here at the Undercroft, we'll be focusing on installing and configuring Pi-hole on a Pi!

- Any model RPi is suitable for this application.
- Make sure to choose a quality microSD card (*Sandisk*, *Samsung*, etc.)
- A good quality *power supply*!

The Achilles heel of an RPi is microSD card failure due to poor quality power supplies, not being able to deliver consistent power.

Download Raspbian

Grab the official *Raspbian image*. This can be installed directly using `dd` on GNU/Linux or BalenaEtcher on Windows. Otherwise *NOOBS* can be used if you're not comfortable with those methods.



Verify Checksum

Compare the hash of your downloaded Raspbian zip archive of choice, with the SHA-256 sum listed for your specific version. They should match.

Windows:

```
C:\> certutil -hashfile <raspbian archive> SHA256
```

GNU/Linux:

```
$ sha256sum <raspbianarchive>
or
$ shasum -a 256 <raspbianarchive>
```

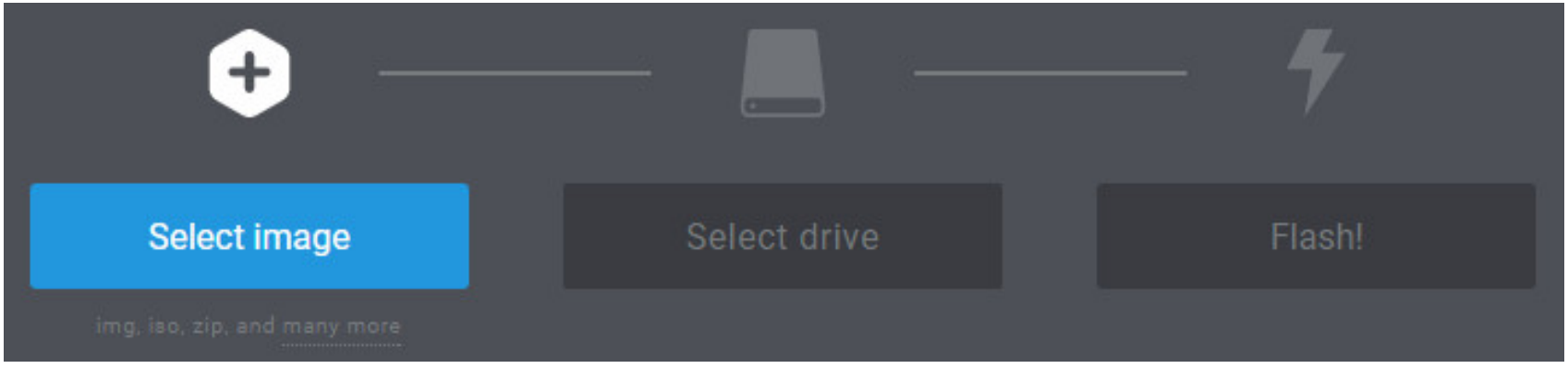
SHA-256:
12ae6e17bf95b6ba83beca61e7394e7411b45eba7e6a520f434b0748ea7370e8

Raspbian image (Windows)

If needed, unzip the archive to get your Raspbian image.

Windows:
It's recommended to use *Raspberry Pi Installer* or *balenaEtcher* to write Raspbian to your MicroSD card.

Note that if you're using a USB SD card reader, to keep an eye out for any driver related issues that may cause the image to not be written properly. If your Pi doesn't boot, this could be why.



Raspbian image (GNU/Linux)

GNU/Linux:
Consult the *documentation* for more details!

- Insert your microSD card and find it's device name:

```
$ dmesg | tail
```

```
[ 4166.226228] mmc0: new high speed SDXC card at address e624
[ 4166.228357] mmcblk0: mmc0:e624 SR128 119 GiB
[ 4166.241647]  mmcblk0: p1 p2
```

```
$ lsblk -p
```

Check to see if any partitions have been mounted. You can unmount them with the below command before continuing.

```
$ sudo umount /dev/<devicename>p1
```

Raspbian image (GNU/linux)

```
$ sudo dd bs=4M if=<raspbianimage> of=/dev/<devicename>
conv=fsync status=progress
```

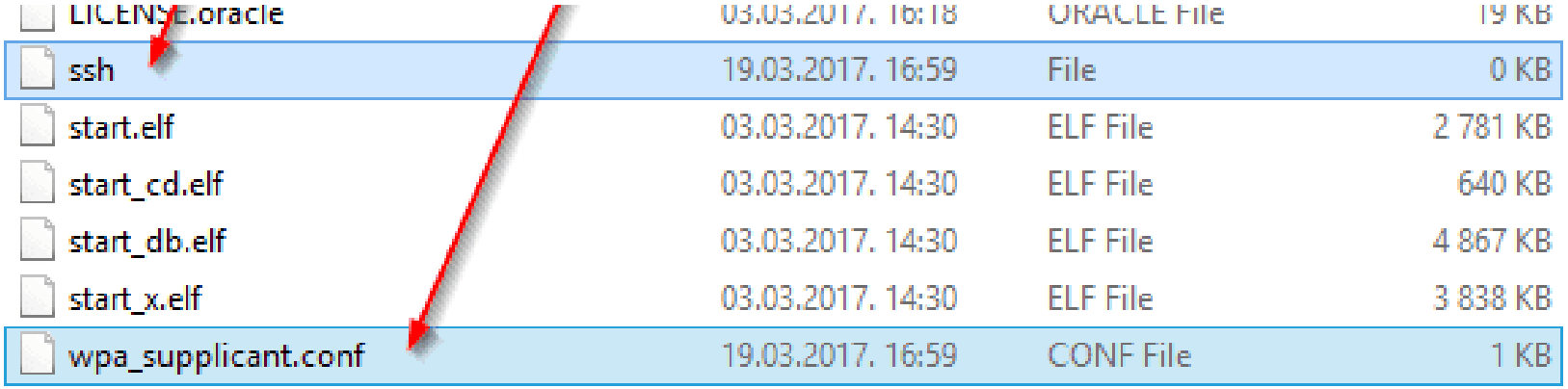
```
NAME
       dd - convert and copy a file

SYNOPSIS
       dd [OPERAND]...
       dd OPTION

DESCRIPTION
       Copy a file, converting and formatting according to
the operands.
```

Always make sure the output file is the microSD card and not your hard drive!

Headless boot

If you chose the Raspbian image with the Desktop Environment and have access to a display, you can connect via HDMI and boot it up. Otherwise, you'll need to do a headless boot using SSH.

- Add a file with the filename of, "`ssh`" to the microSD. If you're on Windows you can create a new text file. Just make sure no file extension is added!



Wireless boot config

For wireless:

- In addition you'll need to create a file called "*wpa_supplicant.conf*" and add information about the wireless AP you want to connect to.

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=US

network={
      ssid="Your network name/SSID"
      psk="Your WPA/WPA2 security key"
      key_mgmt=WPA-PSK
}
```

Finding your Pi

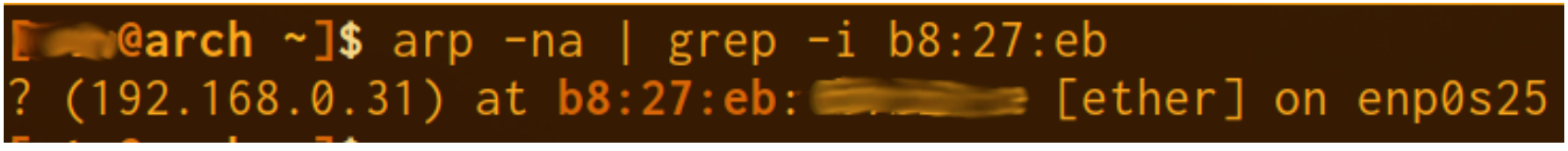
Once you have the Pi booted up, you'll need to find it's IP address. This can be done via your routers client section or by a simple network range scan. If the Pi has populated your ARP tables already, you could try the following commands:

Windows:

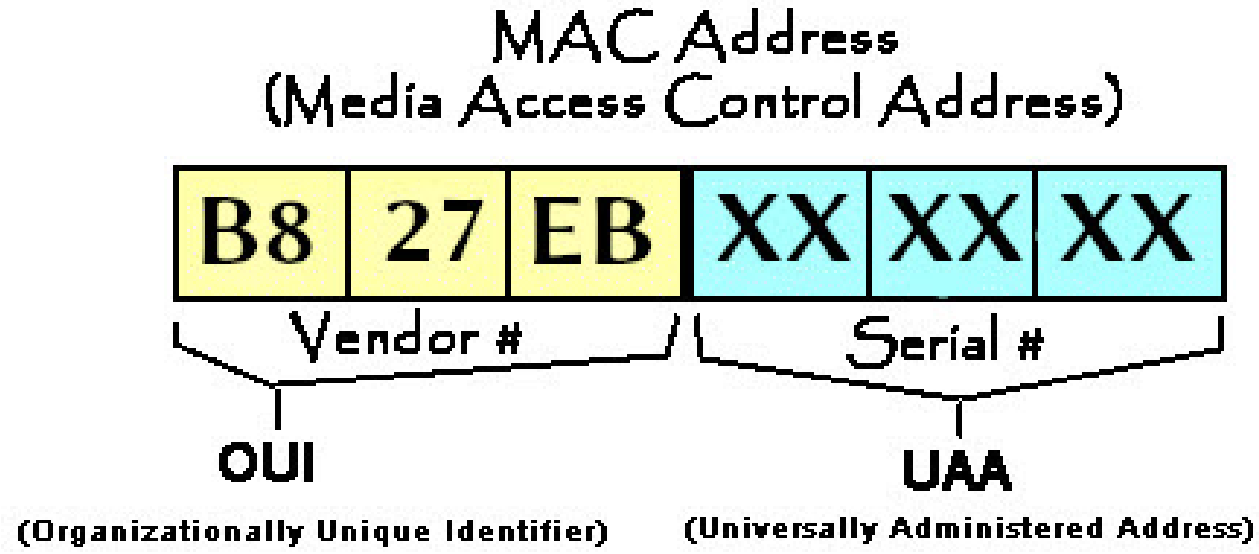
```
C:\> arp -a | findstr b8-27-eb
```

GNU/Linux:

```
$ arp -na | grep -i b8:27:eb
```



What is b8:27:eb?



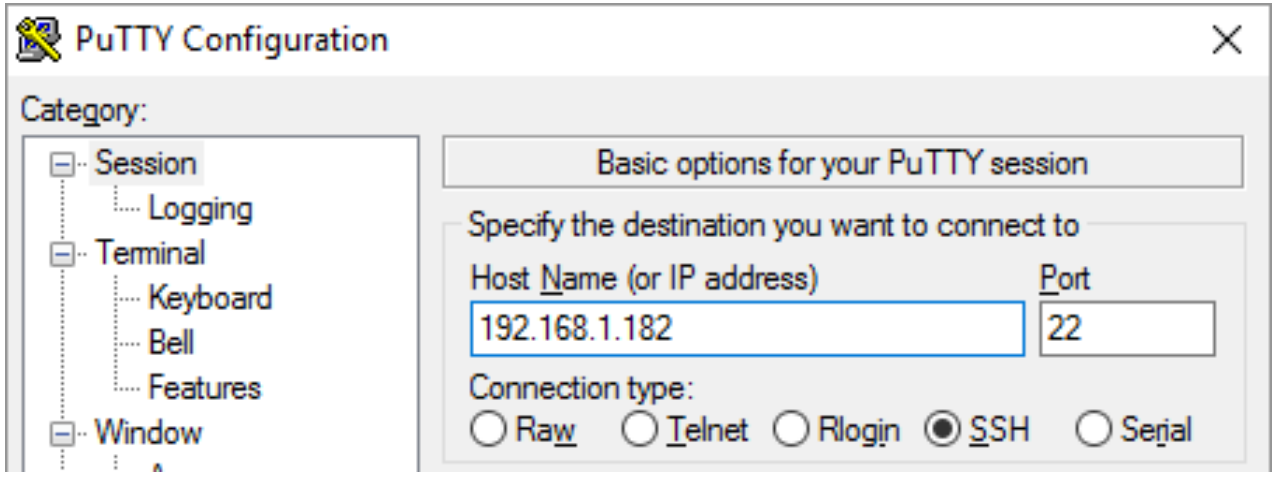
ARP (Address Resolution Protocol) is used to map network addresses to hardware addresses (MAC). In this case, it'll help us find our network neighbors. The first 3 octets represent the OUI, which are unique to the Manufacturer. The Raspberry Pi Foundation's OUI for models up to the RPi 3 B+ is B8:27:EB. The RPi 4 now uses an OUI of DC:A6:32, due to a subsidiary being formed for trading and engineering activities.

SSH

Once you've found the RPi's IP address you can SSH into it. On Windows, the utility *PuTTY* can be used. Otherwise issue the command below on GNU/Linux.

```
$ ssh pi@<ipaddress>
```

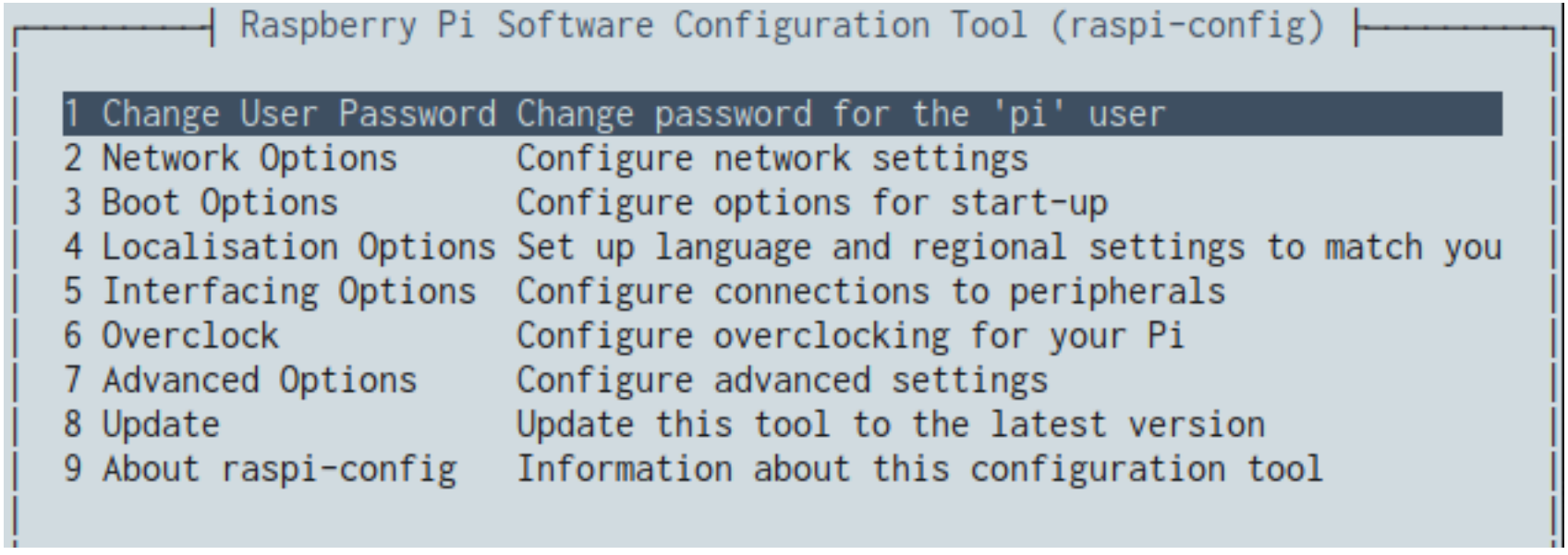
- default password: raspberrypi or raspberry



raspi-config

Once SSH'd in, I highly recommend changing the default password! The hostname and other options can be configured here easily as well. You can do so with the following command:

```
$ sudo raspi-config
```



Pi-hole installation script

The automated *install script* is very well commented! This makes it easy to take a peek and understand what it's doing under the hood, especially if you're new to Bash or shell scripting in general.

```
# A simple function that just echoes out our logo in ASCII format
# This lets users know that it is a Pi-hole, LLC product
show_ascii_berry() {
  echo -e "
        ${COL_LIGHT_GREEN}.;;,.
        .ccccc:,.
         :cccclll:.      ..,,
          :ccccclll.   ;ooodc
           'ccll:;ll .oooodc
             .;cll.;;looo:.
                ${COL_LIGHT_RED}.. ','.
               .',,,,,,'.
             .',,,,,,,,,,.
           .',,,,,,,,,,,,....
         ....''',,,,,,,'.......
        .........  ....  .........
        ..........      ..........
        ..........      ..........
        .........  ....  .........
          ......,,,,,,,'.......
            ....',,,,,,,,,,,.
               .',,,,,,,,,'.
                .',,,,,,'.
                  ..'''.${COL_NC}
"
}
```

Downloading & Executing

Never curl/wget a script from the internet and pipe directly into bash, without first examining the code! *Here's one reason why.*

Method 1:

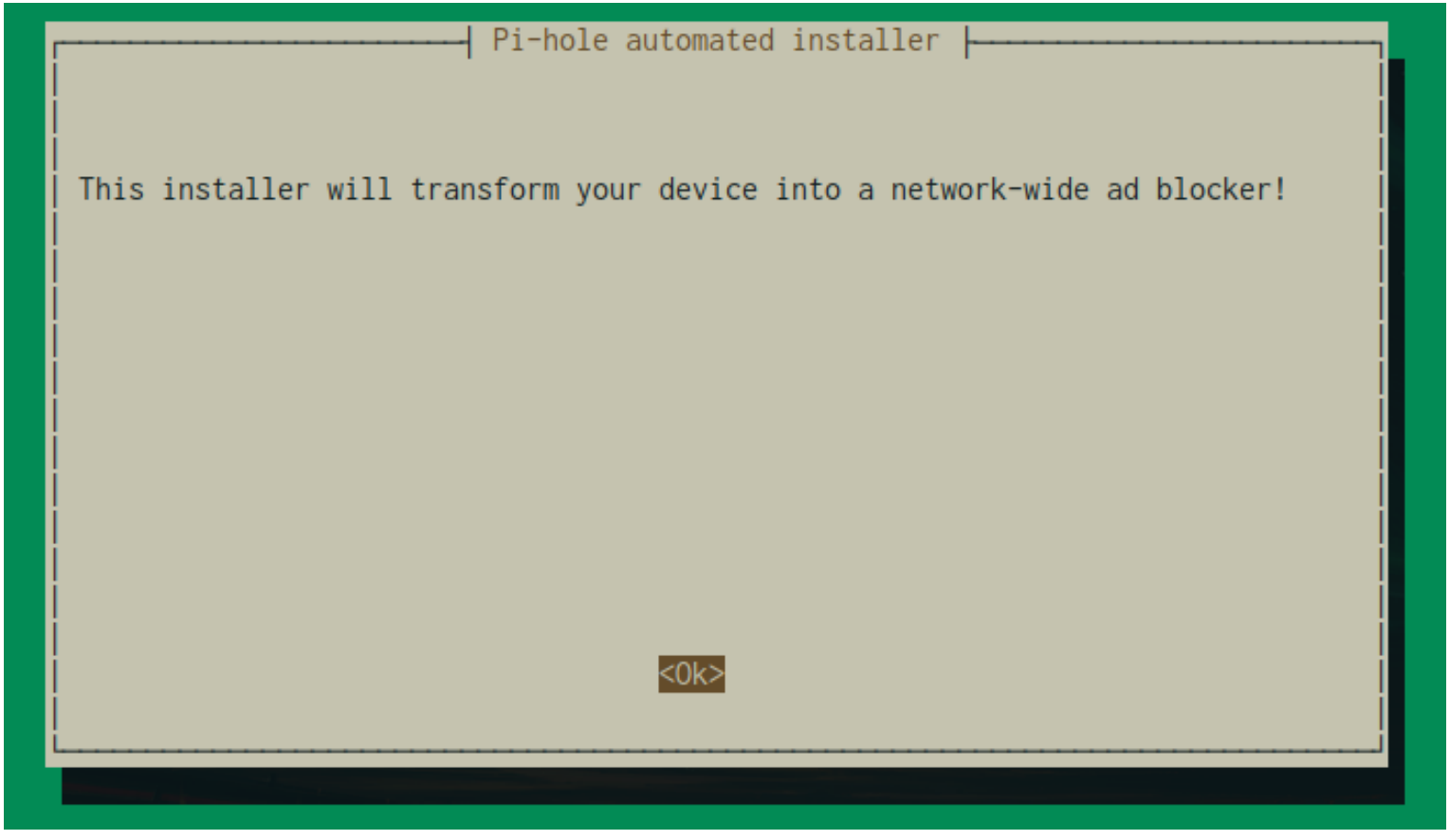
```
$ git clone --depth 1 https://github.com/pi-hole/pi-hole.git Pi-hole
$ cd "Pi-hole/automated install/"
$ sudo bash basic-install.sh
```

Method 2:

```
$ wget -O basic-install.sh https://install.pi-hole.net
$ sudo bash basic-install.sh
```

Automated installer

Follow the on-screen prompts by using your arrow keys to highlight and space bar to select options. Pressing Enter/return key will move to the next page, when <ok> is highlighted or you can exit with <cancel>.

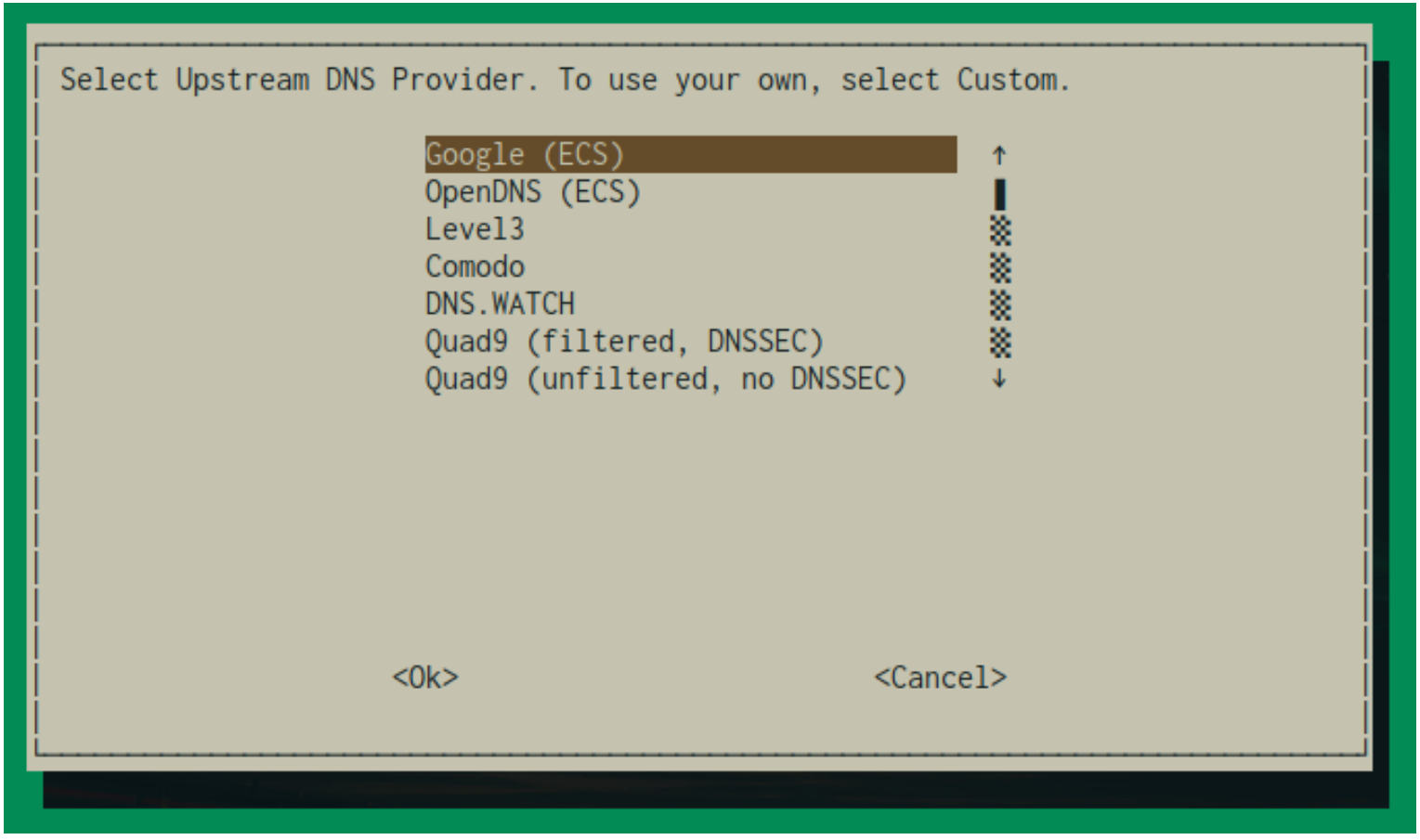


Choose an Interface

Select the interface you'll be using on the RPi. eth0 will be your ethernet interface and wlan0 is your wireless.

Upstream provider

This option refers to who receives the forwarded DNS query from your Pi-hole. You can input a custom provider as well.



Upstream provider comparison

There are a ton of different *DNS providers* out there. You'll find ones tailored for *threat blocking*, adult content and more.

CloudFlare DNS

CloudFlare will never log your IP address (the way other companies identify you). The independent DNS monitor DNSPerf ranks Cloudflare's DNS the fastest DNS service in the world.

Quad9

Quad9 is a free, recursive, anycast DNS platform that provides end users robust security protections, high-performance, and privacy.

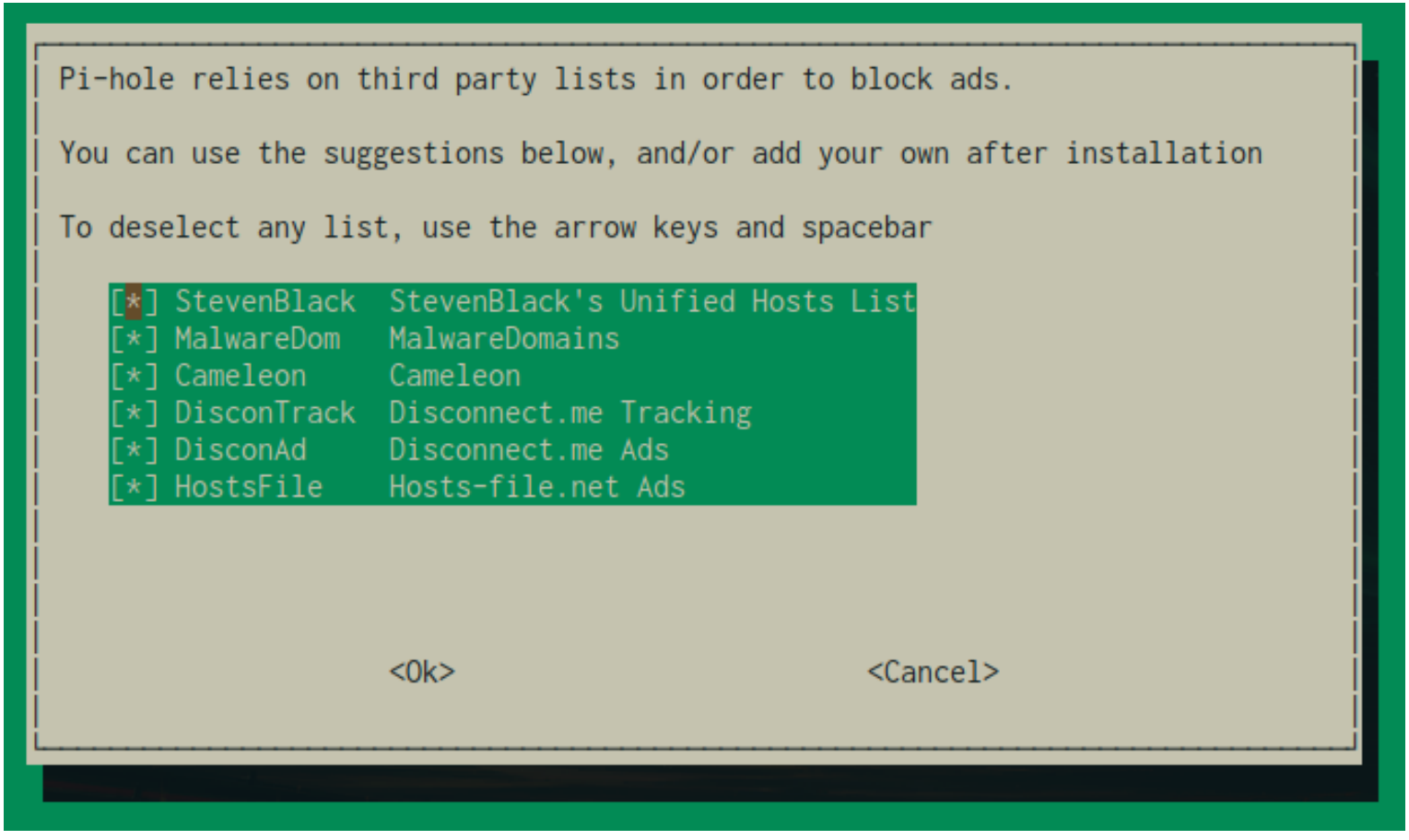
OpenDNS Home (owned by Cisco)

Built-in features include a phishing filter, this is the OpenDNS version the Pi-hole would use if you select it during setup.

You may even decide you want to run your own recursive resolver on the Pi, which we'll cover in a bit!

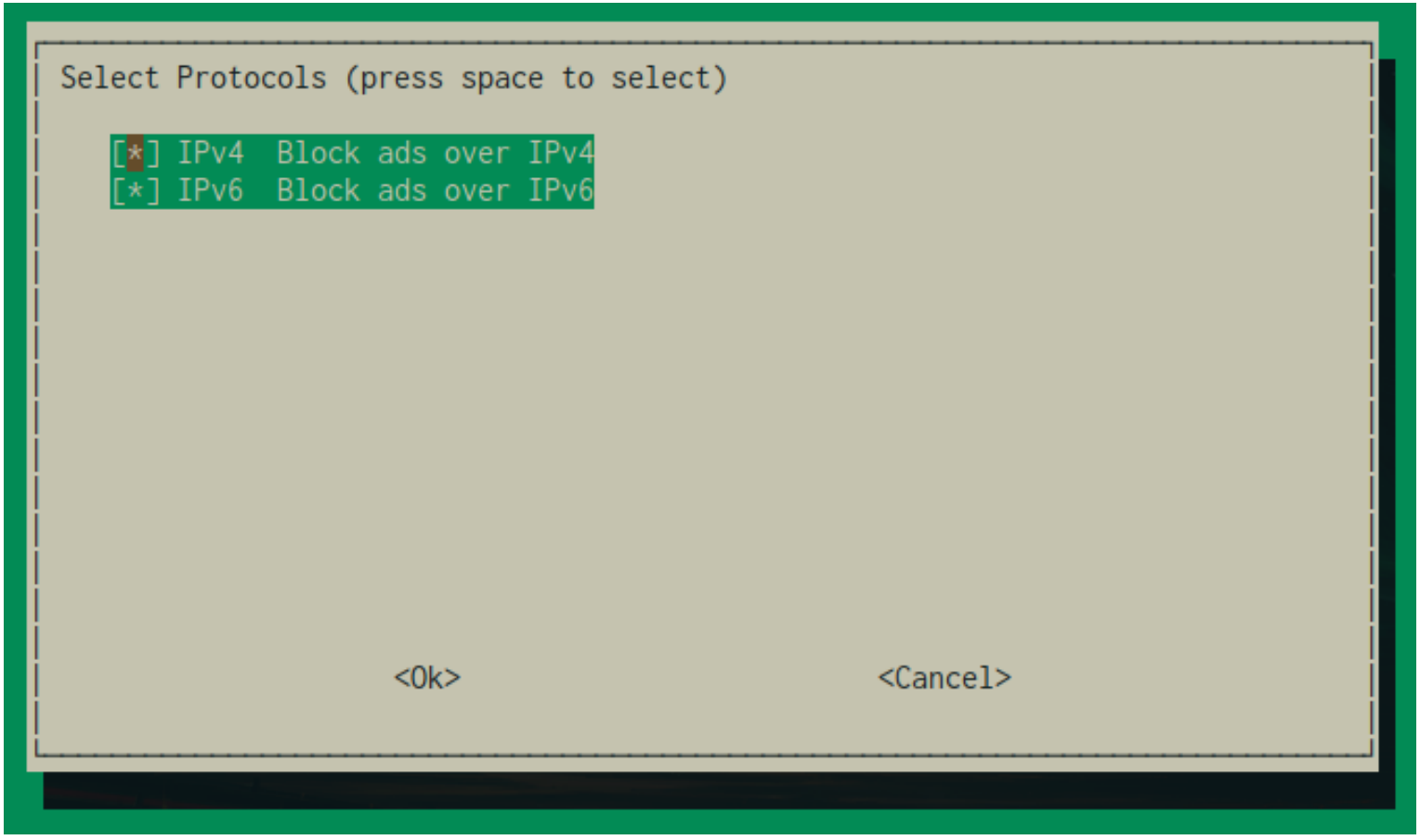
Blocklists

I typically choose all of the preset blocklists during installation. We can add more after installation.



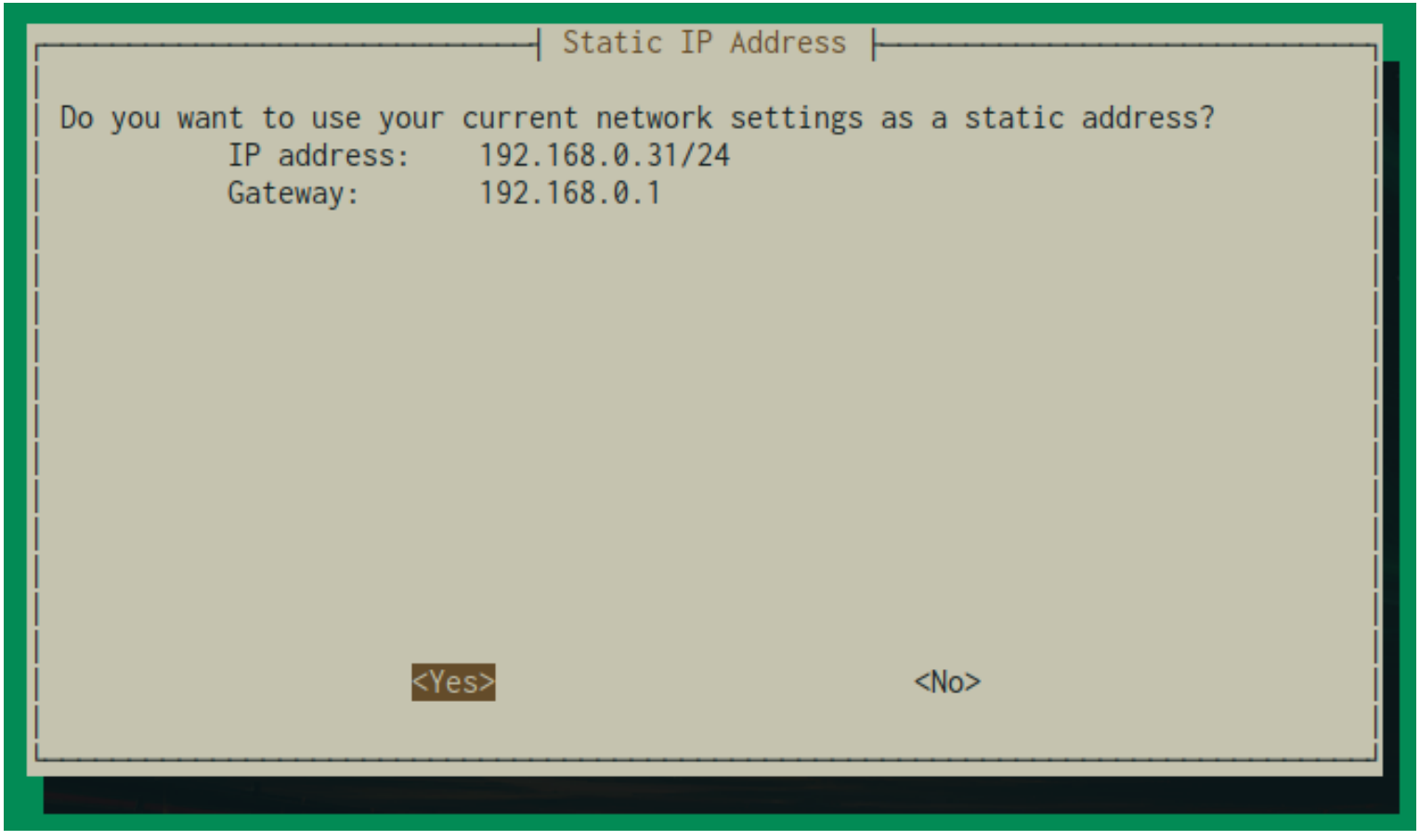
IPv4 & IPv6

You can leave both chosen as the default. It won't have any adverse effects leaving IPv6 selected. It will just generate an IPV6 address for the Pi.



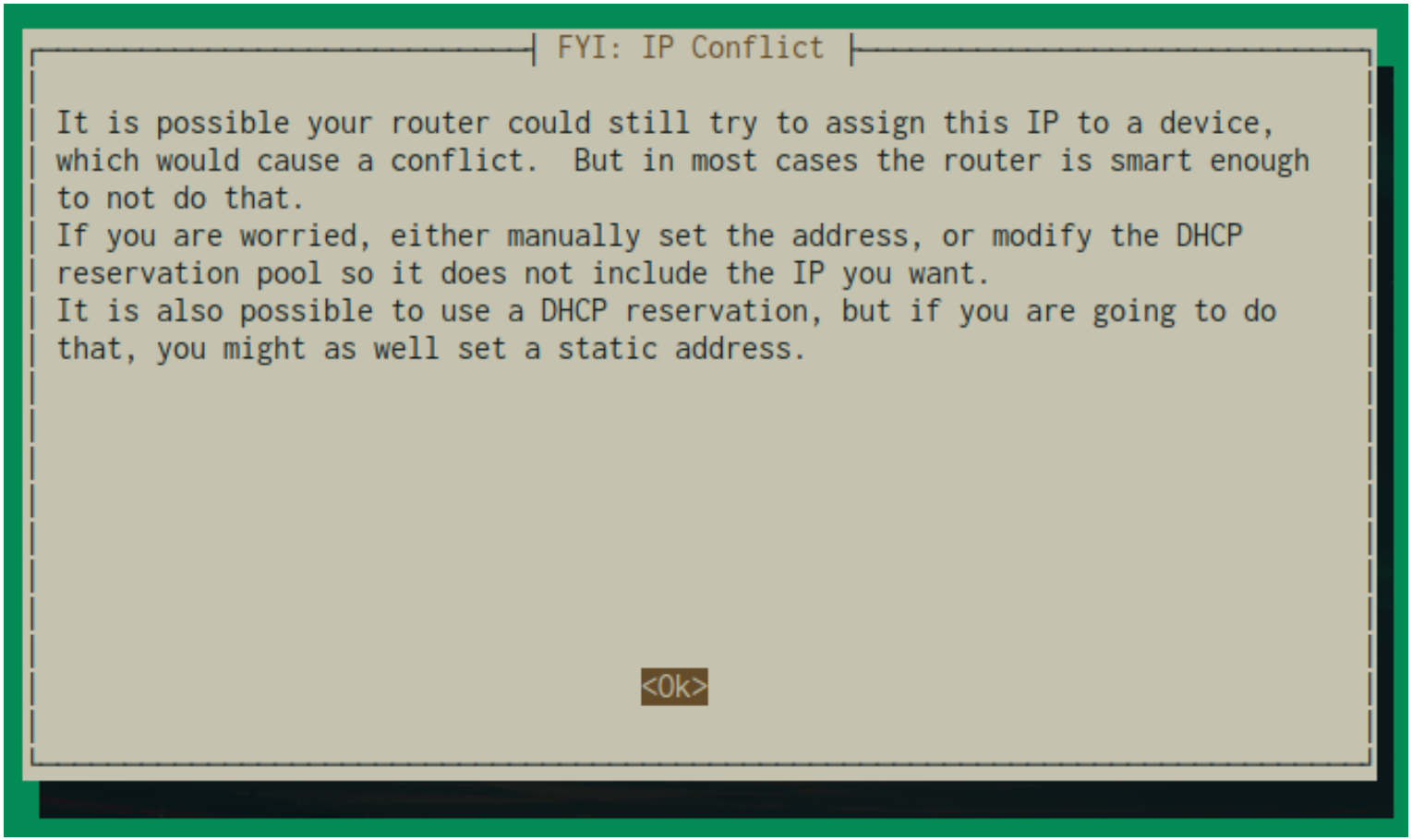
Static IP

Since this is a *server*, we'll need it's location to remain constant on the network.



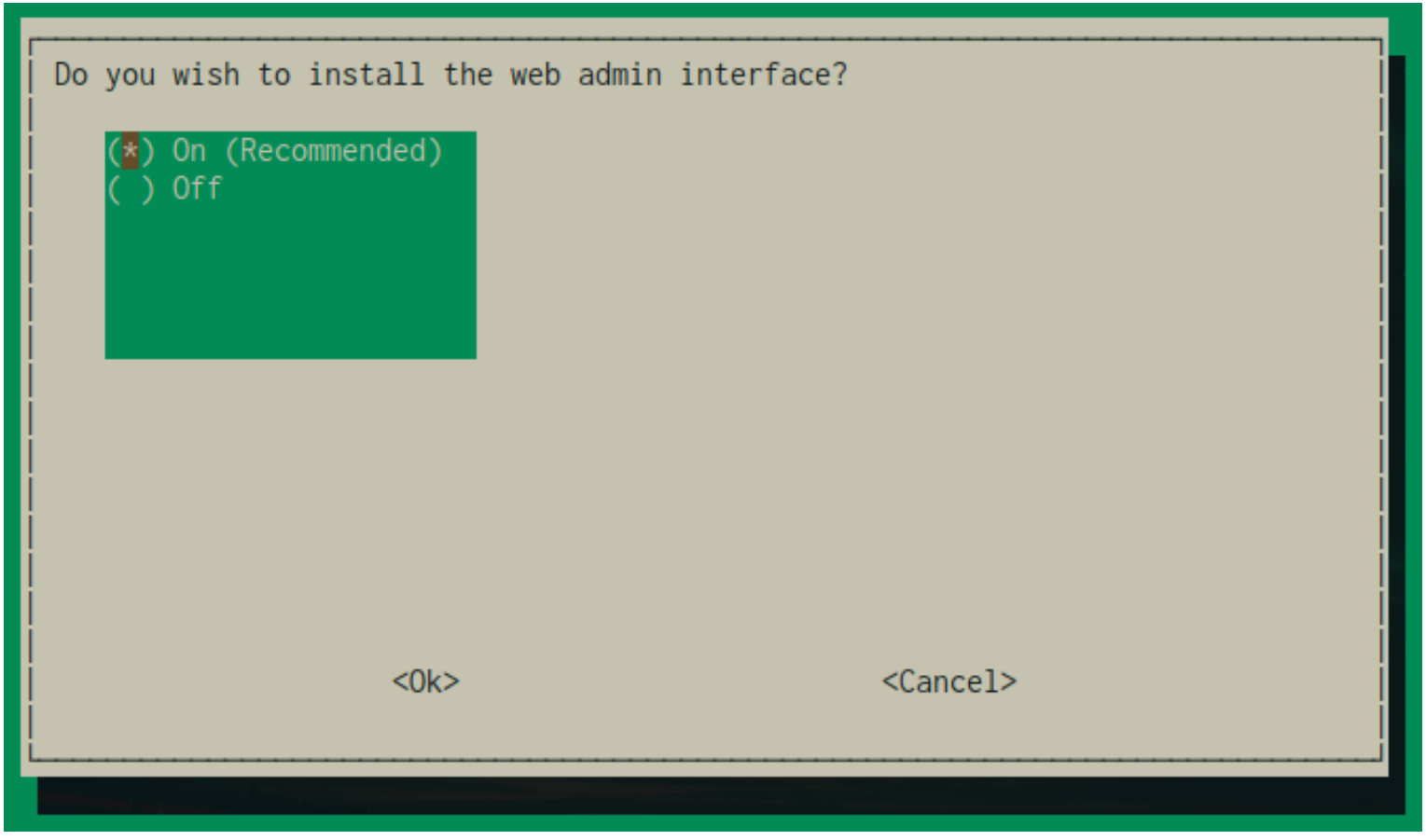
IP conflict

If the static IP is in your DHCP pool, you can easily set a reservation based on the MAC address of the Pi. This is done within your router's interface. Although this usually doesn't present an issue.



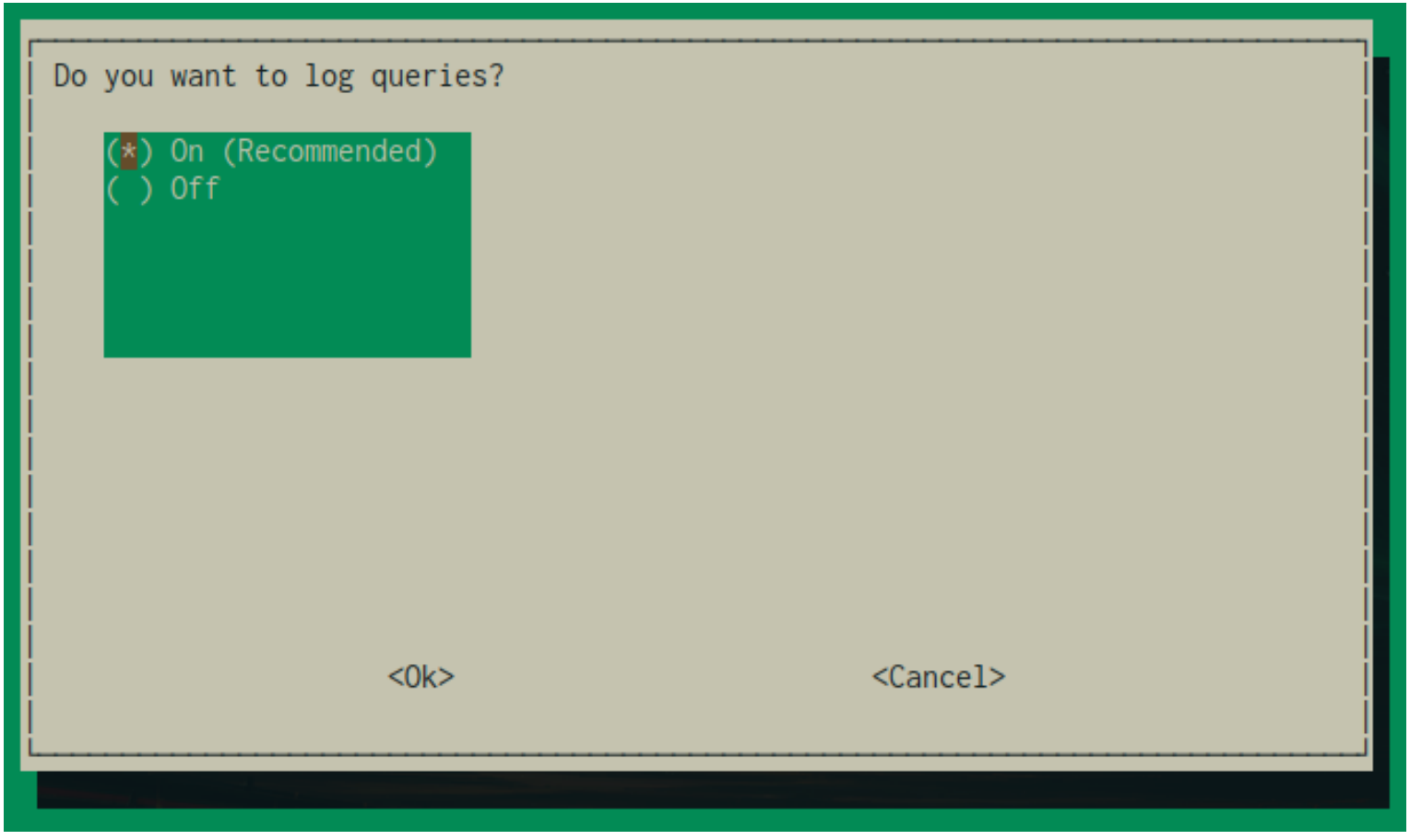
Web admin interface

Yes we want the web admin interface! This will allow us to easily make changes or configure from any device on the network.



Log queries

Yes we want to log queries. This will allow us to see historic trends and more.



FTL privacy mode

This determines the level of information given with statistics. You can read more on this *here*.

Installation complete!

This pages reviews your info, along with giving you the password for the admin interface. Make sure to write this down. It can be changed with the command `$ pi-hole -a -p`, later.

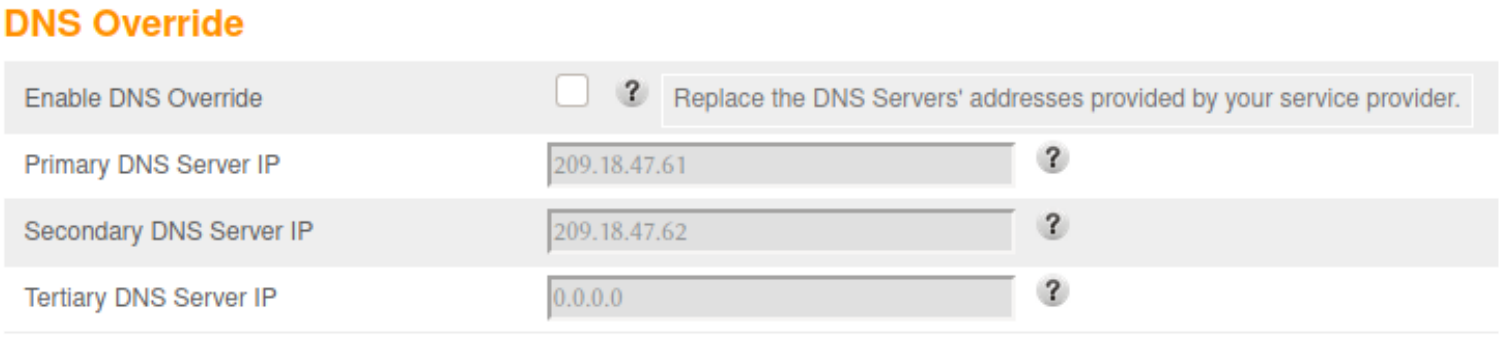
What now?

Pi-hole is ready to go but now you need to make sure clients on the network use Pi-hole. There's two main methods for doing this and they'll depend on your hardware.

- **Network-wide method (2 *options*)**
  - *Option 1*
    - Change the DNS server on your router.
  - *Option 2*
    - Disable DHCP on your router and use the Pi-hole's DHCP server.
- **Per-device method**
  - Manually configure each device to use Pi-hole as their DNS server.

Network-wide method (Option 1)

This involves changing the DNS server from within your router's admin panel. If you have an ISP router, this may not be an option as they sometimes have hardcoded DNS. In that case, you'll want to use Option 2.

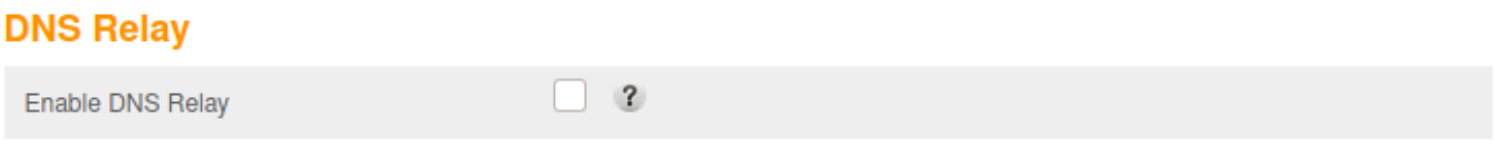


It's also important that you don't use a secondary DNS! This defeats the purpose of the Pi-hole. If however you're forced to input a secondary, try using the Pi-hole's address for both.

Network-wide method (Option 1)

Another possible disadvantage to using this option, is that (depending on the router) all requests on the Pi-hole may appear as if they come from the router's IP. This can be annoying as part of the fun with Pi-hole is being able to differentiate various clients.

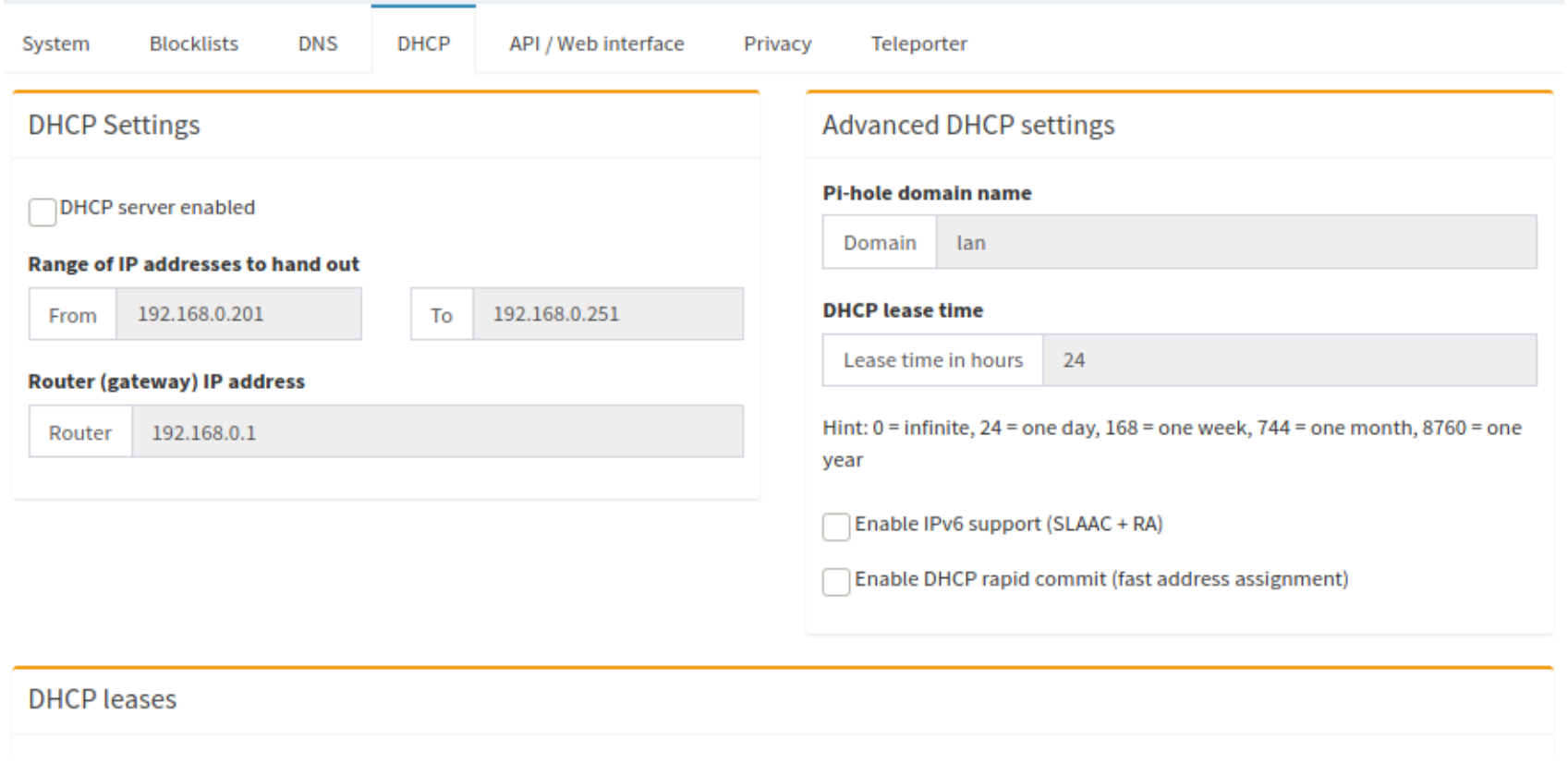
- Try looking for a DNS relay option or something similar and disabling it.



Note that the DNS change will not propagate until DHCP leases are renewed. Restarting your devices should usually force a renewal.

Network-wide method (Option 2)

If the above methods don't suit your hardware, the best option is to disable DHCP on your router and use the Pi-hole's. This capability is present due to Dnsmasq.



Network-wide method (Option 2)

Head over to "*<ipaddress>/admin*" and choose the settings page from the navigation bar. Under the DHCP tab, the server can be enabled, along with a myriad of other options. Don't forget to disable DHCP on your router!

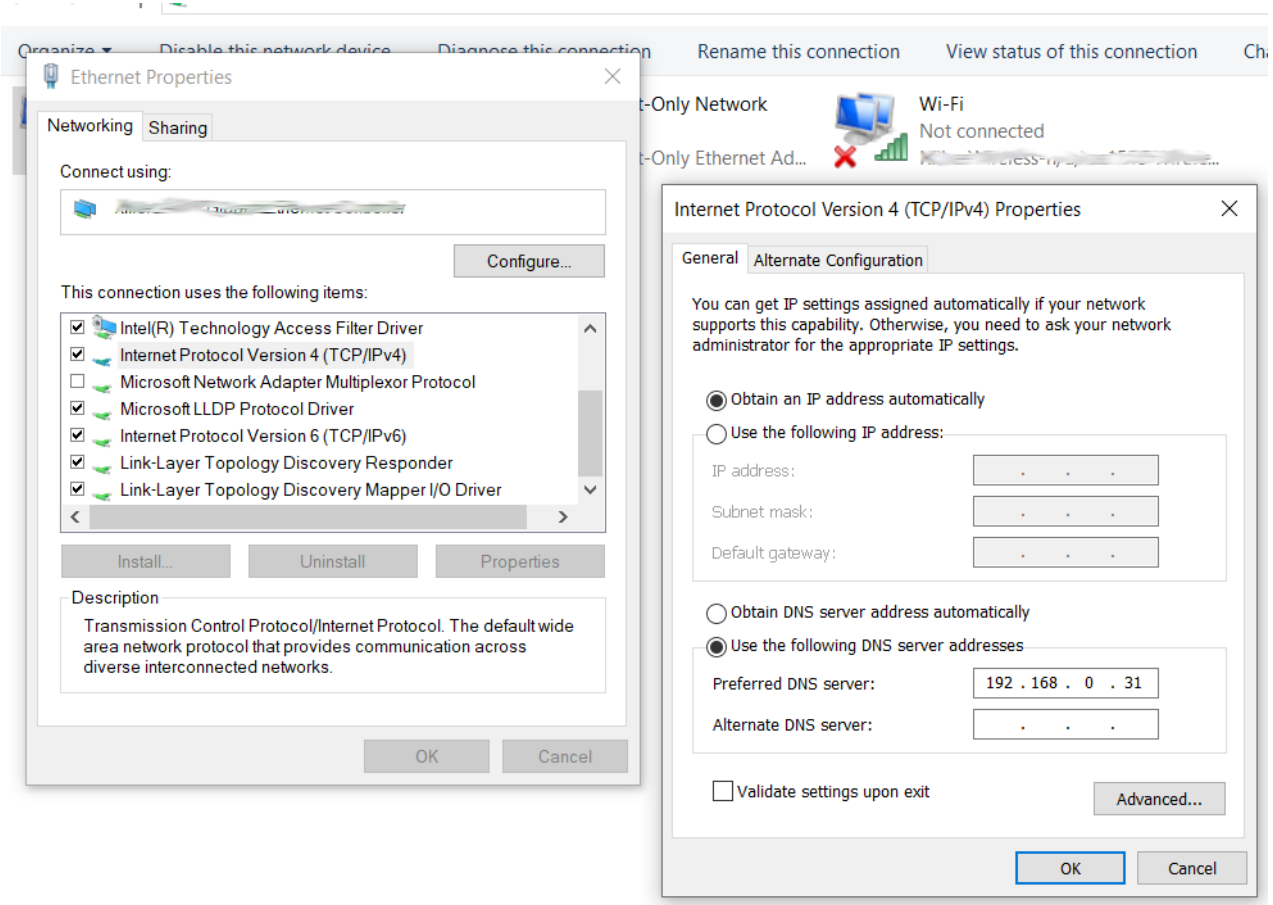
The options should be pre-filled but if not, edit them to your specifics.

- Your router's IP address
- DHCP address lease range (e.g. 192.168.0.2 - 192.168.0.254)

In addition to solving many of the issues outlined earlier. You get the added benefit of having hostnames present for better client tracking!

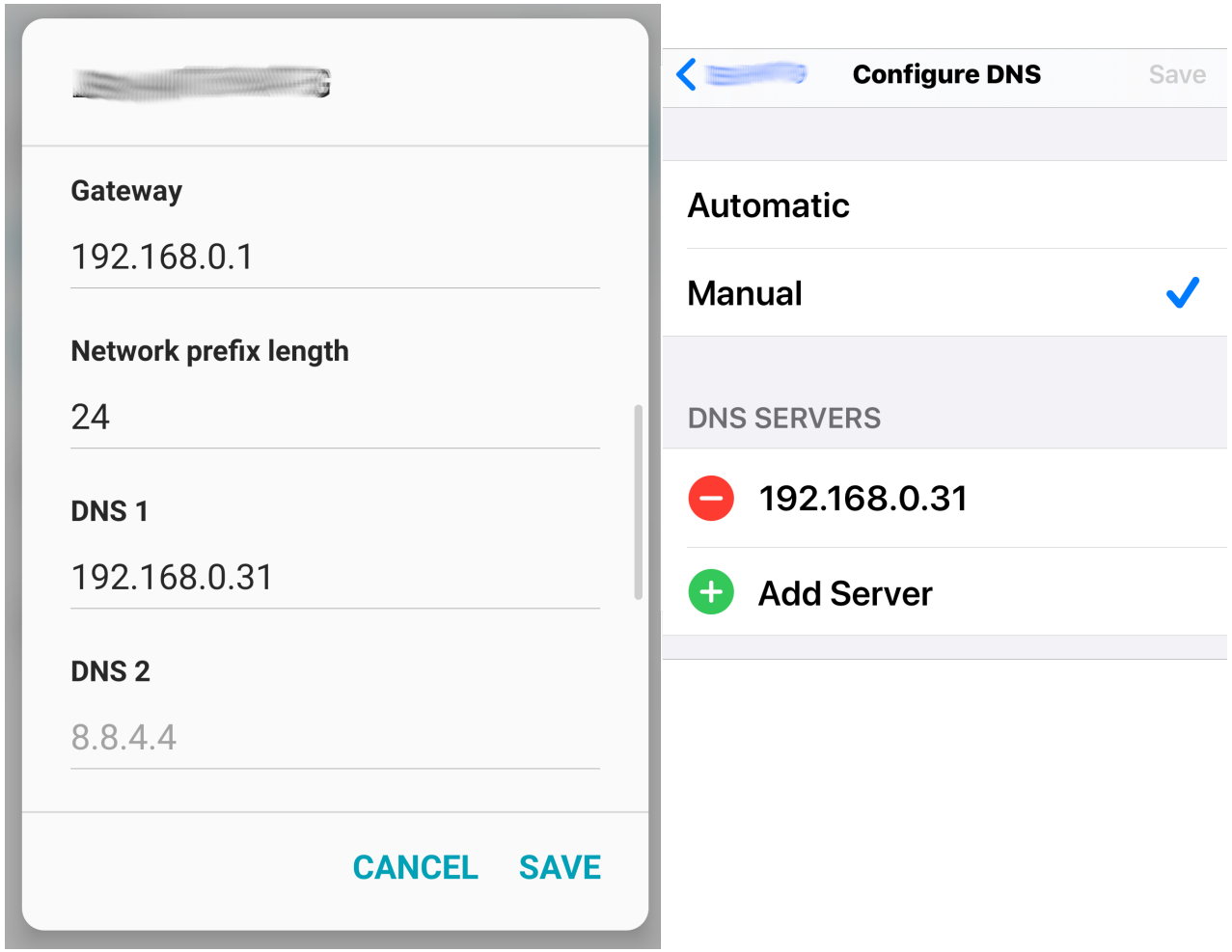
Per-device method

If you only want certain devices to utilize Pi-hole as their DNS server or maybe just want to play around with Pi-hole before making any network-wide changes, you could choose this method.



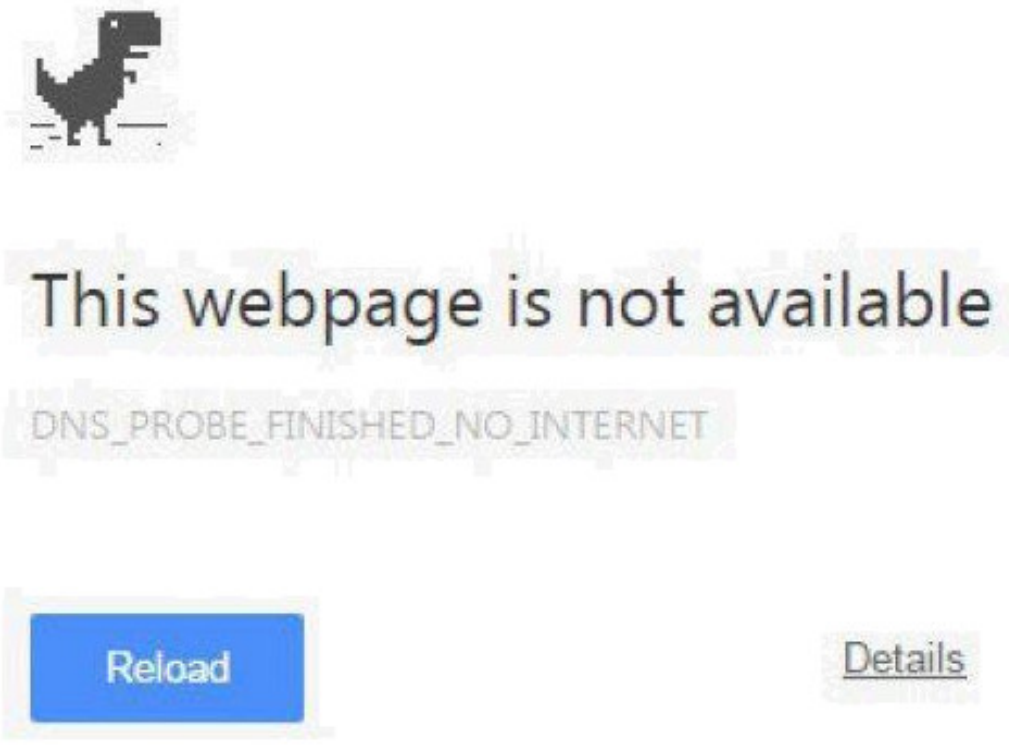
Per-device method

On mobile devices these options can typically be found under more advanced wireless settings.



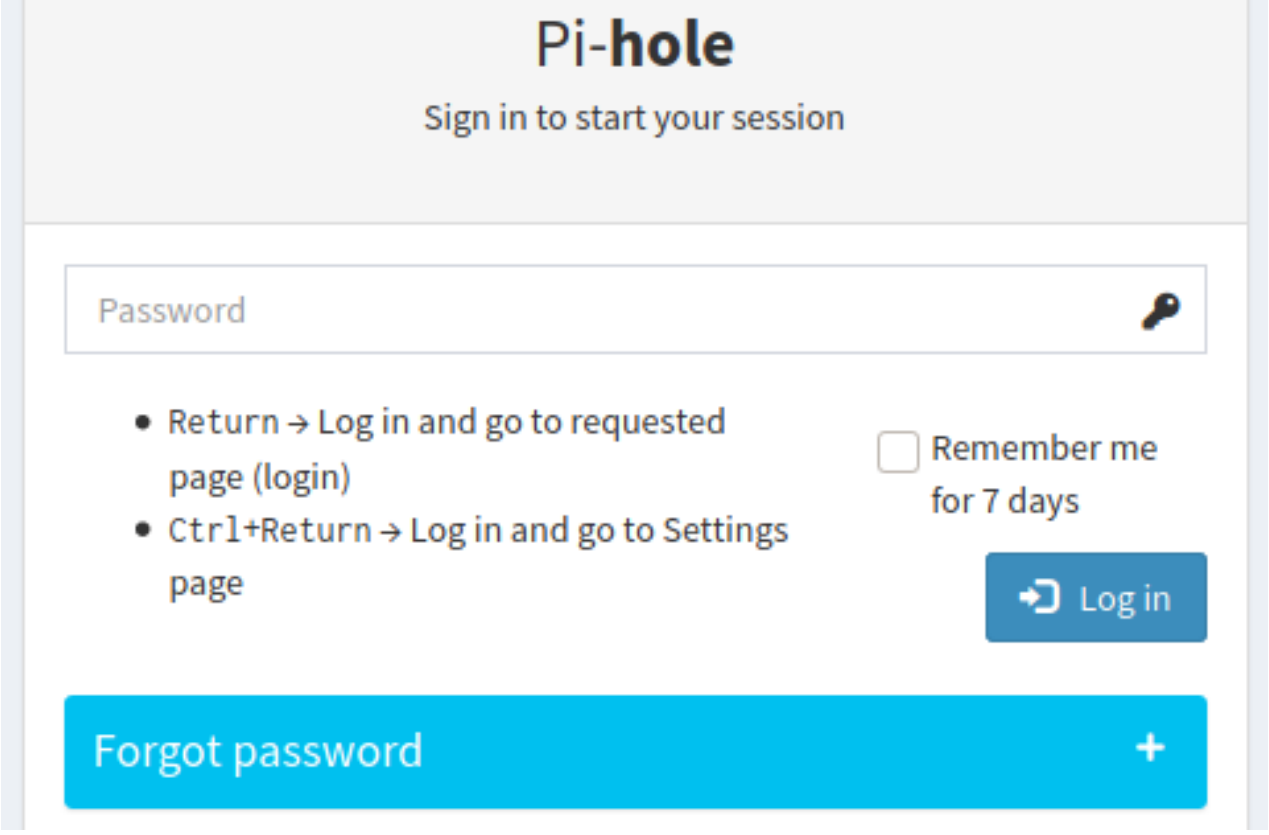
Per-device method

The biggest disadvantage particularly on something like Windows, when using this method is that once you leave your network, you'll have to reconfigure your DNS settings!

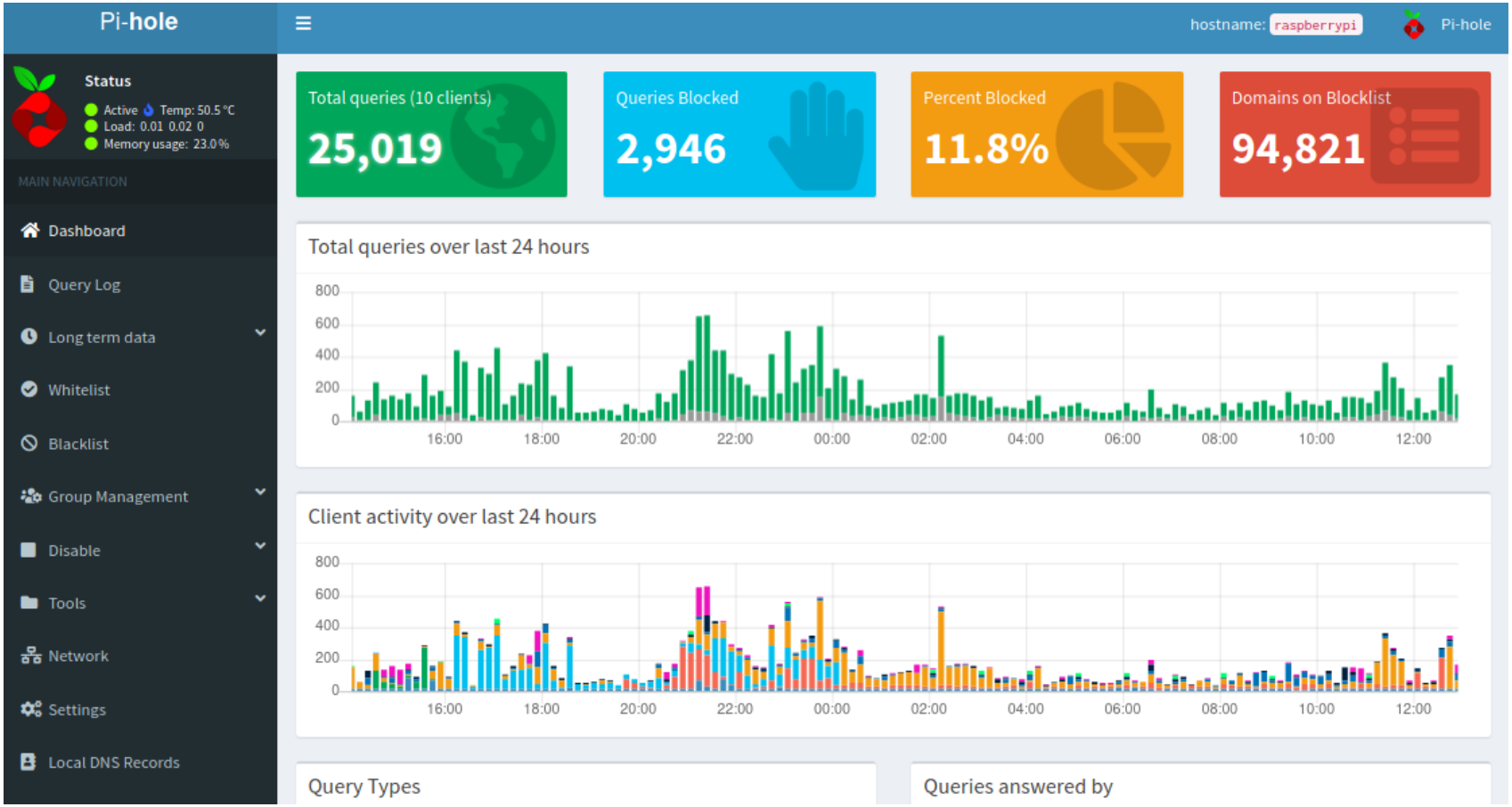


Pi-hole web interface

Now that we're using Pi-hole as our DNS we can start to explore the web interface. Navigate over to *pi.hole/admin* or "*<ipaddress>/admin*". Login using the admin password generated during install.



At-a-glance


Pi-hole
hostname: raspberrypi
Pi-hole
Status
Active Temp: 50.5 °C
Load: 0.01 0.02 0
Memory usage: 23.0%
MAIN NAVIGATION
Dashboard
Query Log
Long term data
Whitelist
Blacklist
Group Management
Disable
Tools
Network
Settings
Local DNS Records
Total queries (10 clients)
25,019
Queries Blocked
2,946
Percent Blocked
11.8%
Domains on Blocklist
94,821
Total queries over last 24 hours
800
600
400
200
0
16:00
18:00
20:00
22:00
00:00
02:00
04:00
06:00
08:00
10:00
12:00
Client activity over last 24 hours
800
600
400
200
0
16:00
18:00
20:00
22:00
00:00
02:00
04:00
06:00
08:00
10:00
12:00
Query Types
Queries answered by

Dashboard

On the dashboard, you can see various Pi-hole statistics:

- Summary: A summary of statistics showing how many total DNS queries have been blocked today, what percentage of DNS queries have been blocked, and how many domains are in the compiled ad list. This summary is updated every 10 seconds.
- Queries over time: Graph showing DNS queries (total and blocked) over 10 minute time intervals. More information can be acquired by hovering over the lines. This graph is updated every 10 minutes.
- Query Types: Identifies the types of processed queries
- Forward Destinations: Shows to which upstream DNS the permitted requests have been forwarded to.
- Top Domains: Ranking of requested sites by number of DNS lookups.
- Top Advertisers: Ranking of requested advertisements by number of DNS lookups.
- Top Clients: Ranking of how many DNS requests each client has made on the local network.

The Top Domains and Top Advertisers lists may be hidden depending on the privacy Settings on the settings page

Note that the login session does *not* expire on the dashboard, as the summary is updated every 10 seconds which refreshes the session.

Header

Header

Top left: Status display

Shows different status messages:

- Status: Current status of the Pi-hole - Active (🟢), Offline (🔴), or Starting (🟠)
- Temp: Current CPU temperature
- Load: load averages for the last minute, 5 minutes and 15 minutes, respectively. A load average of 1 reflects the full workload of a single processor on the system. We show a red icon if the current load exceeds the number of available processors on this machine (which is 4)
- Memory usage: Shows the percentage of memory actually blocked by applications. We show a red icon if the memory usage exceeds 75%

Top right: About

- GitHub: Link to the Pi-hole repository
- Details: Link to Jacob Salmela's blog with some more details, describing also the concept of the Pi-hole
- Updates: Link to list of releases
- Update notifications: If updates are available, a link will be shown here.
- Session timer: Shows the time remaining until the current login session expires.

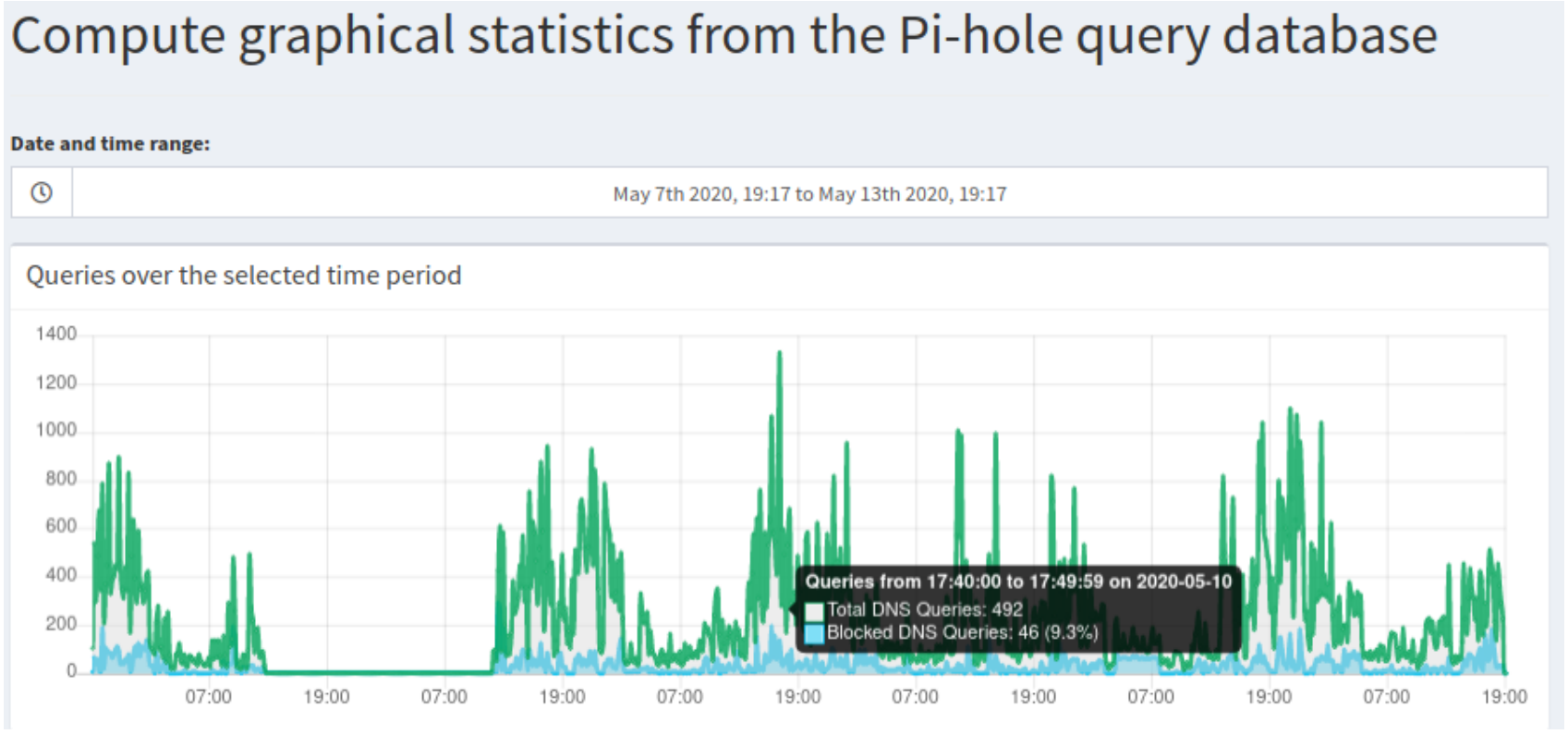
Query log

Shows the recent queries by parsing Pi-hole's log. It is possible to search through the whole list by using the "Search" input field. If the status is reported as "OK", then the DNS request has been permitted. Otherwise ("Pi-holed") it has been blocked. By clicking on the buttons under "Action" the corresponding domains can quickly be added to the white-/blacklist. The status of the action will be reported on this page. By default, only the recent 10 minutes are shown to enhance the loading speed of the query log page. All domains can be requested by clicking on the corresponding link in the header of the page. Note that the result heavily depends on your privacy settings (see Settings page).

Time	Type	Domain	Client	Status	Reply	Action
2020-05-13 18:33:35	AAAA	shareasale.com	192.168.0.15	Blocked (gravity)	- (0.2ms)	Whitelist
2020-05-13 18:33:35	A	shareasale.com	192.168.0.15	Blocked (gravity)	- (0.2ms)	Whitelist
2020-05-13 18:33:19	AAAA	telemetry.malwarebytes.com	192.168.0.10	Blocked (gravity)	- (0.4ms)	Whitelist
2020-05-13 18:33:19	A	telemetry.malwarebytes.com	192.168.0.10	Blocked (gravity)	- (0.6ms)	Whitelist
2020-05-13 18:32:20	AAAA	shareasale.com	192.168.0.15	Blocked (gravity)	- (0.4ms)	Whitelist
2020-05-13 18:32:20	A	shareasale.com	192.168.0.15	Blocked (gravity)	- (0.5ms)	Whitelist
2020-05-13 18:33:25	AAAA	www.eff.org	192.168.0.15	OK (forwarded)	CNAME (88.7ms)	Blacklist
2020-05-13 18:33:25	A	www.eff.org	192.168.0.15	OK (forwarded)	CNAME (64.8ms)	Blacklist
2020-05-13 18:33:24	AAAA	snippets.cdn.mozilla.net	192.168.0.15	OK (forwarded)	CNAME (178.4ms)	Blacklist

Long-term data

From here you can see historic trends in the form of graphs, query logs and top lists, by choosing date and time ranges.



Whitelist/Blacklist

Add or remove domains (or subdomains) from the white-/blacklist. If a domain is added to e.g. the whitelist, any possible entry of the same domain will be automatically removed from the blacklist and vice versa.

Regex blacklisting is supported (entering `^example` will block any domain starting with `example`, see also our Regex documentation). You can still whitelist specific domains even if they fall under a regex pattern.

You can white-/blacklist multiple entries at a time if you separate the domains by spaces.

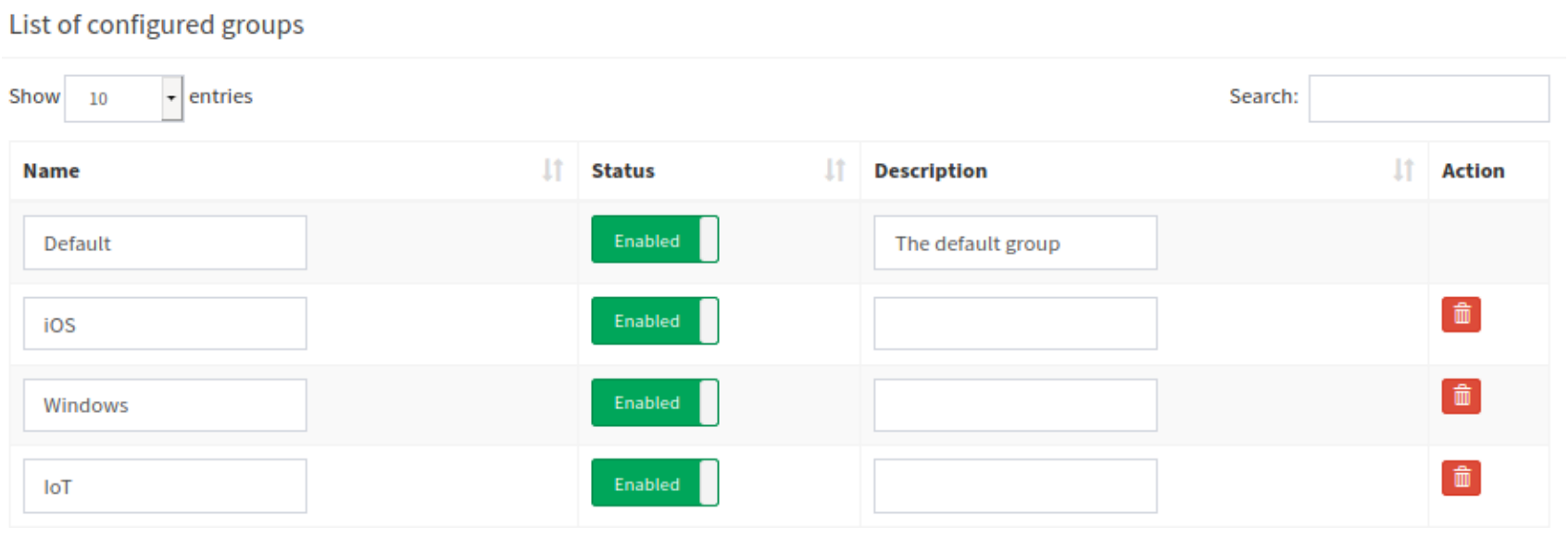
Domain/RegEx	Type	Status	Comment	Group assignment	Action
spclient.wg.spotify.com	Exact whitelist	Enabled	Migrated from /etc/pihole/	Default	
apresolve.spotify.com	Exact whitelist	Enabled	Migrated from /etc/pihole/	Default	
googleapis.l.google.com	Exact whitelist	Enabled	Migrated from /etc/pihole/	Default	
secure.netflix.com	Exact whitelist	Enabled	Migrated from /etc/pihole/	Default	
api-global.netflix.com	Exact whitelist	Enabled	Migrated from /etc/pihole/	Default	
appboot.netflix.com	Exact whitelist	Enabled	Migrated from /etc/pihole/	Default	
adeventtracker.spotify.com	Exact whitelist	Enabled	Migrated from /etc/pihole/	Default	
connectivitycheck.gstatic.com	Exact whitelist	Disabled	Migrated from /etc/pihole/	Default	

Blacklist

Domain/RegEx	Type	Status	Comment	Action
`*.gstatic.com`	Regex blacklist	Enabled	Migrated from /etc/pihole/	
`^(.+[_.-])?ad[sxv]?[0-9]*[_.-]`	Regex blacklist	Enabled	Migrated from /etc/pihole/	
`^(.+[_.-])?adse?rv(er?|ice)?s?[0-9]*[_.-]`	Regex blacklist	Enabled	Migrated from /etc/pihole/	
`^(.+[_.-])?telemetry[_.-]`	Regex blacklist	Enabled	Migrated from /etc/pihole/	
`^(www[0-9]*\.)?xn--`	Regex blacklist	Enabled	Migrated from /etc/pihole/	
`^adim(age|g)s?[0-9]*[_.-]`	Regex blacklist	Enabled	Migrated from /etc/pihole/	
`^adtrack(er|ing)?[0-9]*[_.-]`	Regex blacklist	Enabled	Migrated from /etc/pihole/	
`^advert(s|is(ing|ements?))?[0-9]*[_.-]`	Regex blacklist	Enabled	Migrated from /etc/pihole/	
`^aff(iliat(es?|ion))?[_.-]`	Regex blacklist	Enabled	Migrated from /etc/pihole/	
`^analytics?[_.-]`	Regex blacklist	Enabled	Migrated from /etc/pihole/	

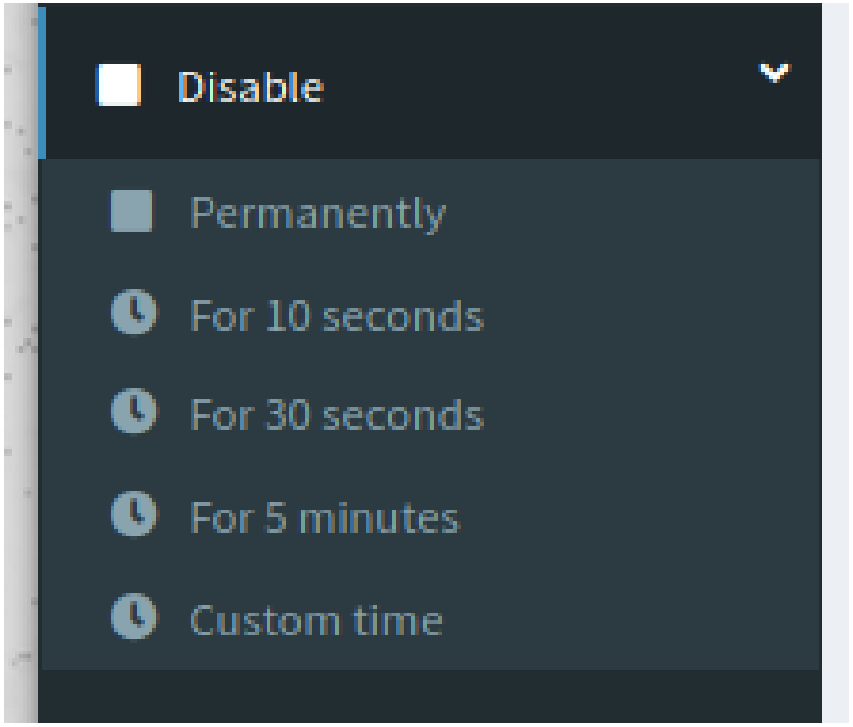
Group management!

New feature added in V5.0. Clients can be added to *groups* here. Adlists are only managed here now, instead of within the settings tab. Domain specifics can be configured here as well.



Disable blocking

If you need to disable the blocking capability for a certain period of time, you can choose the presets or set a custom time from the Disable drop-down menu.



Tools

Some of the main tools you may frequently use include:

Tools → Update Gravity

Will download any updates from the third-party blocklists that we source. By default, this command runs once a week via cron (Sunday).

Tools → Query Lists

This function is useful to find out what list a domain appears on. Since we don't control what the third-parties put on the blocklists, you may find that a domain you normally visit stops working. If this is the case, you could run this command to scan for strings in the list of blocked domains and it will return the list the domain is found on. This proved useful a while back when the Mahakala list was adding `apple.com` and `microsoft.com` to their block list.

Tools → Tail pihole.log

Live tailing of the raw Pi-hole log.

In addition, the audit log allows a quick check up on top allowed/blocked domains, to which you can choose to dismiss from appearing on the list or make changes to.

Network

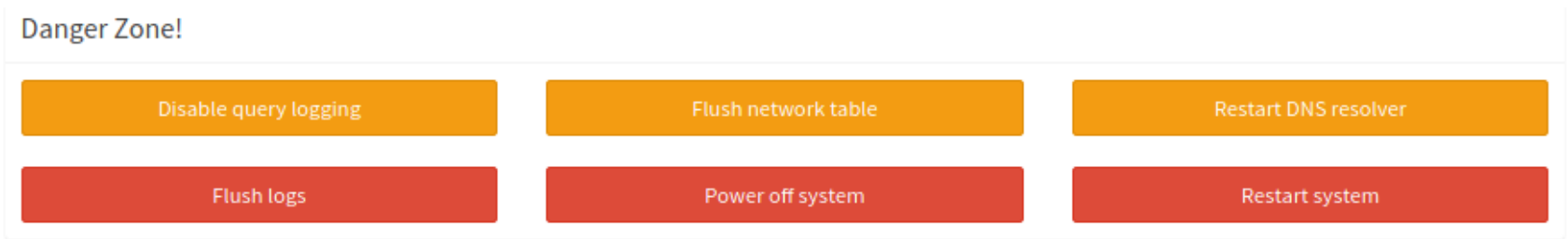
The network tab will give info regarding all clients on the network. MAC address, Hostname (If applicable), when they were first and last seen, as well as if they are using the Pi-hole.

IP address	Hardware address	Interface	Hostname	First seen	Last Query
192.168.0.10	[redacted] Chicony Electronics Co., Ltd.	eth0	N/A	2019-11-11 21:36:00	2020-05-15 14:59:38
192.168.0.9	[redacted] Vizio, Inc	eth0	N/A	2019-11-11 21:18:00	2020-05-15 14:59:15
192.168.0.7	[redacted] Apple, Inc.	eth0	N/A	2019-11-11 21:28:00	2020-05-15 14:57:36
192.168.0.12,2603:9000:6608:1800:50b0:2961:6488:df3f	[redacted] LG Electronics (Mobile Communications)	eth0	N/A	2019-11-11 21:34:00	2020-05-15 14:54:02

Settings

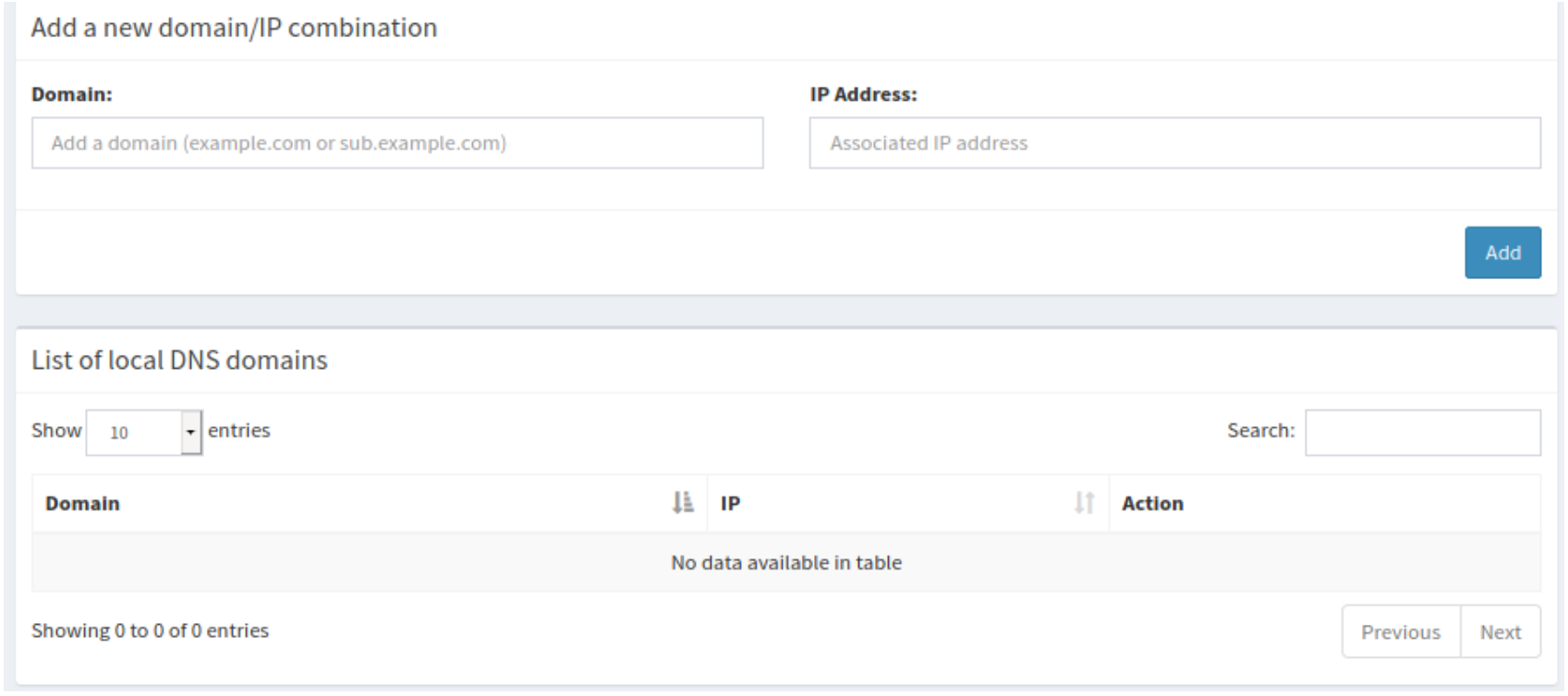
This is where you'll find a variety of admin type options.

- View Network and System info
- Restart/shutdown FTL and system functions
- Manage upstream DNS providers and more
- Enable/manage Pi-hole's DHCP server
- Get your API token
- Change web interface defaults
- Modify FTL privacy levels
- Import/export Pi-hole settings and lists



Local DNS records

This is another new feature in V5.0. It's essentially similar in function to what a "*hosts*" file is used for (i.e /etc/hosts)



Blocklist sources

If you still see some ads or want to block more specific content, you can ad new lists through *Group Management*. A large collection of lists can be seen on *firebog.net*, from a variety of different maintainers. Here are some commonly used regex to start with as well.

The Big Blocklist Collection

The Internet is full of unsavoury content: advertisers wanting to sell you stuff you don't need, trackers extracting and selling your data as if it were oil, and malicious content vying to hijack your favourite device. This collection hopes to help you minimise these issues, and to maintain a more enjoyable online presence, using the wonderful, free and open source utility known as Pi-hole.

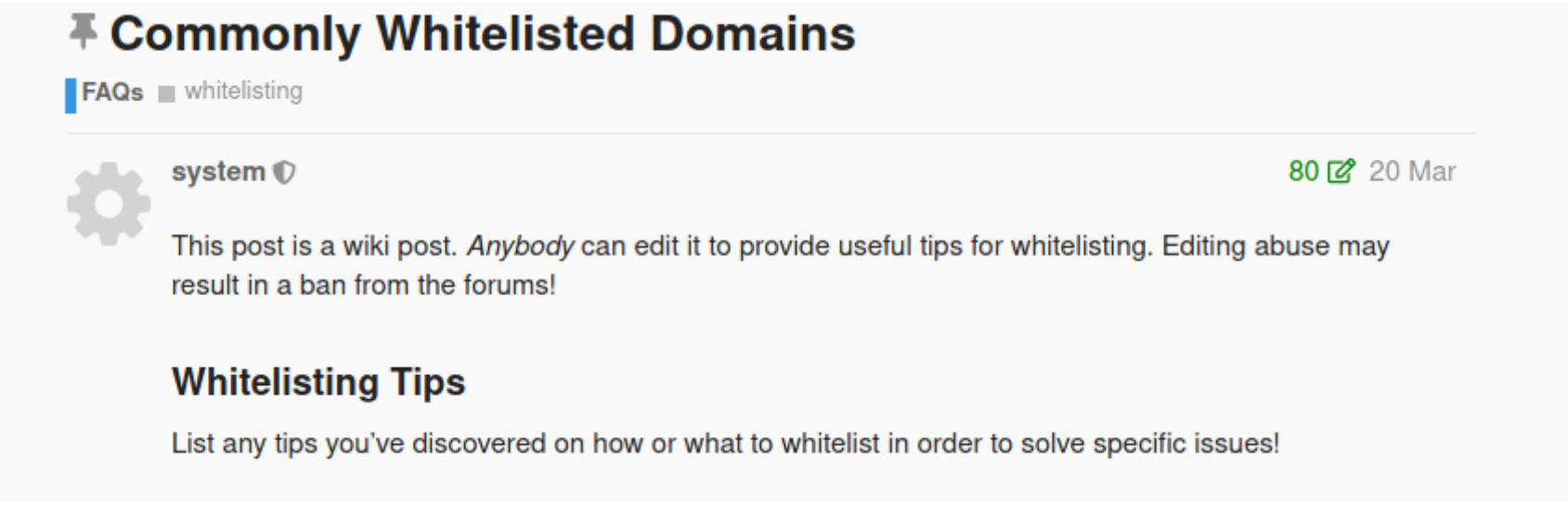
> *On arrival, like a growing number of websites, Forbes asked readers to turn off ad blockers in order to view the article. After doing so, visitors were immediately served with pop-under malware, primed to infect their computers, and likely silently steal passwords, personal data and banking information. Or, as is popular worldwide with these malware "exploit kits," lock up their hard drives in exchange for Bitcoin ransom.*
>
> — *'Forbes Site, After Begging You To Turn Off Adblocker, Serves Up A Steaming Pile Of Malware 'Ads''* - Techdirt, 2016

Before starting, here are some reading points:

✓ Lists bulleted with a tick are least likely to interfere with browsing
✗ Lists bulleted with a cross block multiple useful sites (e.g: Pi-hole updates, Amazon, Netflix)
› A guide on how to add these lists is found here
› If you wish to automate the update of your `adlists.list`, a text-only version is found here
› Using lists hosted at `v.firebog.net` allows me to view very basic ongoing aggregated statistics via CloudFlare
› These lists are painstakingly curated by their respective maintainers. Please contact them *first* if you find false positives
› Avoid using mirrored consolidated lists, if possible; it deprives the original list maintainer of visits (meaning they may be less inclined to keep it up to date!)

Common whitelisting

The more blocklists you add, they higher the chance of false positives. A list of commonly *whitelisted domains* is available and continuously updated on the Pi-hole forums.



Pi-hole from the CLI

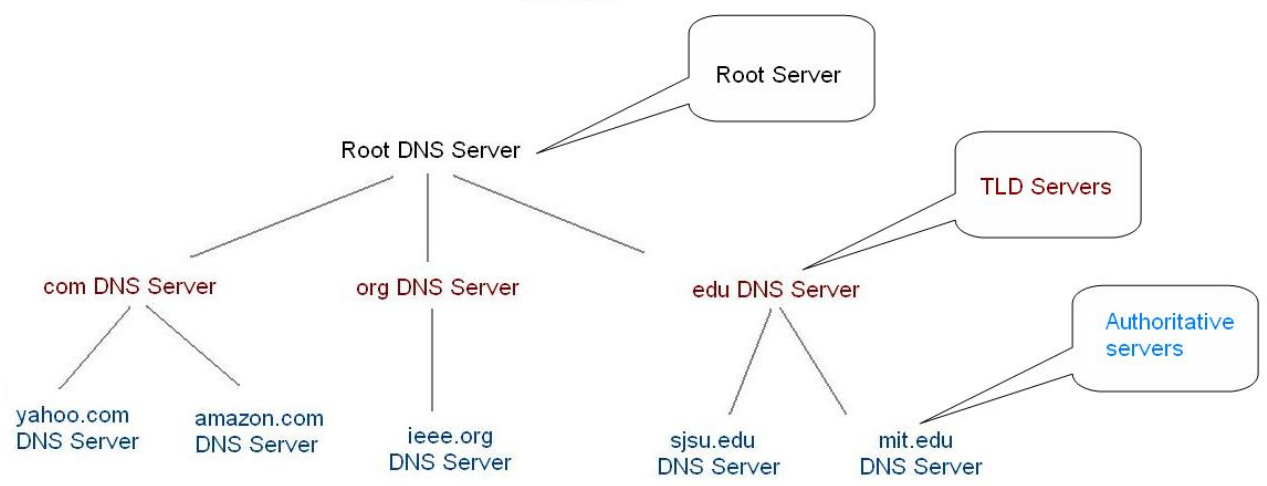
Even though the web interface is intuitive, you may find youself wanting to interact with Pi-hole from the command line.

```
pi@raspberrypi:~ $ pihole
Usage: pihole [options]
Example: 'pihole -w -h'
Add '-h' after specific commands for more information on usage
```

Some common ones include:

- `$ pihole -b <domain>` Add domain(s) to blacklist.
- `$ pihole -w <domain>` Add domain(s) to whitelist.
- `$ pihole -g` Update gravity list.
- `$ pihole -up` Update Pi-hole version.
- `$ pihole restartdns` Restart all subsystems.

Let's talk DNS

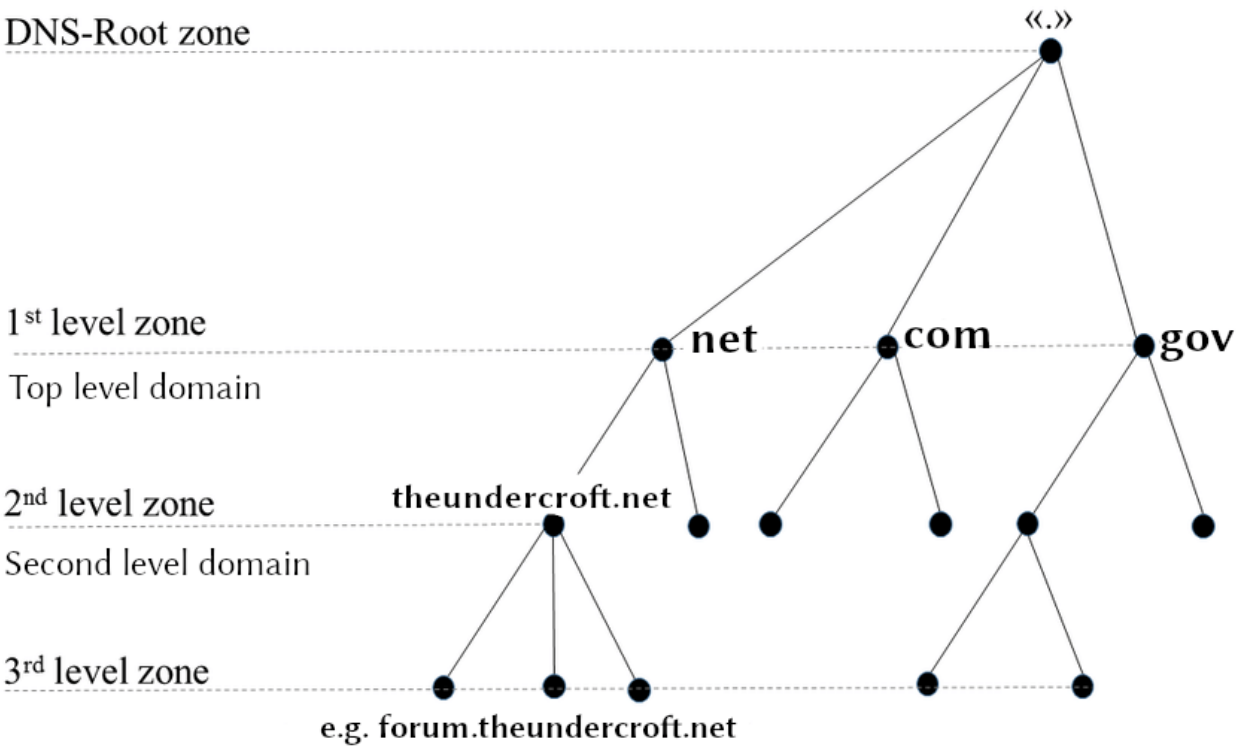


The *Domain Name System* was created and first tested in *1983* as a way to translate (resolve) human-friendly domain names to machine-friendly numbers (i.e. IP addresses). This was in response to the increasingly cumbersome host tables being used at the time.

- Hierarchical
- Decentralized and distributed
- Address changes can occur without effecting end-users

DNS zones

DNS queries that traverse through the domain name space, head through various *zones*. These zones house name servers that contain the trusted records which help to resolve your query or point you in the right direction.



Root zone

The first step a request makes on its journey, is to the root zone (if the browser and OS cache don't have an answer for it). There are 12 entities managing 13 root servers worldwide that are responsible for referring queries to the next step, the TLD (Top Level Domain) zone. This doesn't mean there are only 13 servers though. They're actually distributed amongst hundreds globally.

- Highest level in the hierarchy
- Contains all name and address info for TLDs
- A resolver only needs a *root hints* to reach this zone and start the process

13 Root servers

The *IANA* under *ICANN* is responsible for these root servers. Note that VeriSign manages 2 root servers.

Top Level Domain zone

TLDs are in place to manage domains registered under their respective group/type. There are currently 1500+ TLDs contained within the root zone. These can include everything from your popular *com* or *net*, to country codes such as *ca* or *uk*. The TLD's are managed by *regstries*.

Second-level domain zone

This is the *level* at which you may already be familiar with if you've ever registered a domain name. *Registrars* administer the commercial sale and registration of domains. Not to be confused with Registers that manage the TLDs.

- Authoritative nameservers in this zone maintain the correct records to resolve the IP address.
- The Third-level domain zone is below this one and is another subdomain.

Resource Records

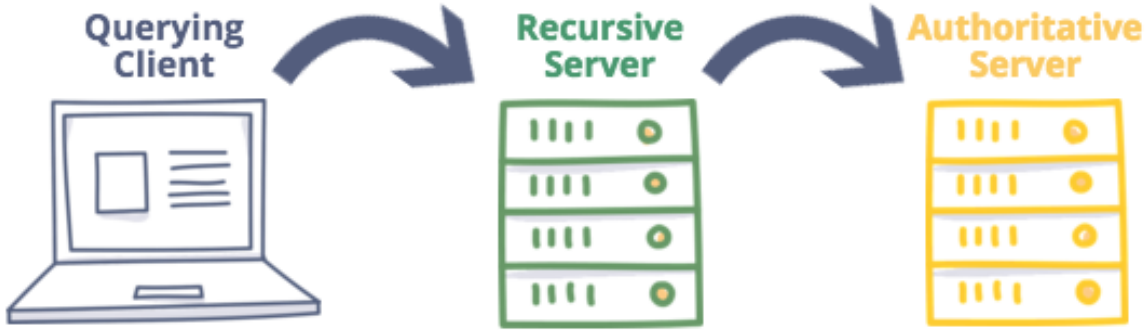
There's a large number of *resource records* that hold various bits of information needed to satisfy certain DNS related requests.

- **Address Mapping record (A Record)**—also known as a DNS host record, stores a hostname and its corresponding IPv4 address.
- **IP Version 6 Address record (AAAA Record)**—stores a hostname and its corresponding IPv6 address.
- **Canonical Name record (CNAME Record)**—can be used to alias a hostname to another hostname. When a DNS client requests a record that contains a CNAME, which points to another hostname, the DNS resolution process is repeated with the new hostname.
- **Mail exchanger record (MX Record)**—specifies an SMTP email server for the domain, used to route outgoing emails to an email server.
- **Name Server records (NS Record)**—specifies that a DNS Zone, such as "example.com" is delegated to a specific Authoritative Name Server, and provides the address of the name server.
- **Reverse-lookup Pointer records (PTR Record)**—allows a DNS resolver to provide an IP address and receive a hostname (reverse DNS lookup).
- **Certificate record (CERT Record)**—stores encryption certificates—PKIX, SPKI, PGP, and so on.
- **Service Location (SRV Record)**—a service location record, like MX but for other communication protocols.
- **Text Record (TXT Record)**—typically carries machine-readable data such as opportunistic encryption, sender policy framework, DKIM, DMARC, etc.
- **Start of Authority (SOA Record)**—this record appears at the beginning of a DNS zone file, and indicates the Authoritative Name Server for the current DNS zone, contact details for the domain administrator, domain serial number, and information on how frequently DNS information for this zone should be refreshed.

Authoritative vs Recursive servers

A recursive DNS server, often called a "resolver" is responsible for traversing the Domain name space on behalf of the client and getting answers from authoritative servers. These authoritative servers hold the correct resource records to point you to the next level until you reach the authoritative name server for the specific domain in which you queried. Thus giving you the IP address.

- Authoritative servers are involved in every step along the way
- Your ISP or Google may be your recursive resolver (think back to upstream providers from earlier)
- You can run your own recursive resolver right on your Pi!



Caching

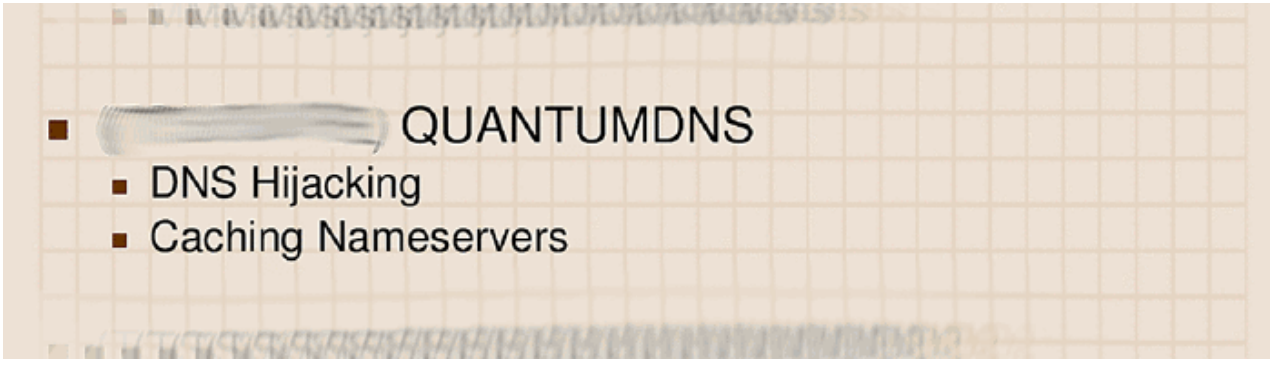
Caching is a form of temporary data persistence. In the case of DNS it is used along every layer to help speed up resolution and reduce load, by saving records in memory. Central to DNS caching is a term called *TTL* (Time to Live). The TTL is essentially an expiration date in seconds for how long a resolver should cache a particular resource record.

- Your browser's cache is the first stop when a query needs to be resolved (TTLs here are often very short).
- Your OS level cache is then checked against the query.
  - If it can't find the requested record, it will leave your local machine and head to your DNS resolver.
- Your resolver will check it's own cache and continue on through the zones until it resolves the query through a name server's cache or you reach the authoritative ns which translates the IP address.

DNS security issues

Even though DNS is one of the oldest concepts pertaining to the modern internet, it hasn't changed much in the past 30+ years since it's inception.

- DNS queries are still sent in **clear text**!
- Lack of authentication
- DNS Hijacking, Poisoning and other *attack vectors* are common
- ISPs and DNS providers typically log queries



Targeting DNS

Man-in-the-middle attacks on DNS

In March of 2014, the government of Turkey decided to block Twitter inside the country. They did so at the DNS level by instructing Turkish ISP's to return specified records for twitter.com and send them to a Turkish government web site. People quickly realized what was happening, and, by using foreign recursive resolvers, they could bypass the restrictions and get continue to get Twitter's true address. This inspired citizens to spray paint Google's public DNS server address on buildings and public spaces.

DNSSEC

DNS security extensions were proposed in response to some of these issues, starting in the early 90s. It works by implementing a digital signing policy that encompasses all layers of DNS.

- Public-key crypto
- Chain of trust built from the root server
- Allows authentication and trust of the records received

Limitations:

Although DNSSEC has been around for sometime now, it is still not fully implemented around the world. Along with this, it merely provides validation of records. The DNS queries are still being sent in clear text.

DNS-over-HTTPS

DoH is something that's stirring the pot as of late. Due in part to *Firefox* pushing it as a default. DoH revolves around sending DNS queries over port 443 with other HTTPS traffic. These go to an upstream provider that supports DoH, in effort to stymie tracking of your queries.

There are some *concerns* surrounding DoH. Aside from obvious enterprise *monitoring issues*, there are data leaks that occur potentially making it's use not very beneficial.

- Destination IP is still seen by ISP
- *SNI (Server name indication)* leaks
  - *Encrypted SNI* is currently in the works to mitigate this
- *OCSP (Online certificate status protocol)* & *CT (Certificate Transparency) log* leaks

DNS-over-TLS

DoT is similar yet different from DoH. As opposed to queries being camouflaged within normal HTTPS (443) traffic on DoH. DoT uses its own port (853) and adds TLS on top of UDP. This allows for better monitoring of DNS in an enterprise environment from a network security perspective, yet still allows for the encryption of DNS queries.

Concerns still exist with regards to centralization of encrypted DNS through a handful of providers due to these methods.

It's important to note that DNSSEC can still be used in conjunction with both DoT & DoH for validation of queries.

Local Recursive Resolver

Unbound is a caching, validating, open source recursive resolver.

There are potential privacy improvements by having your resolver go to the root zone and beyond without a big provider involved. It can often times be hard to trust a provider when they say they don't log queries, etc.

The main downside to Unbound is that recursively traversing the name space for a requested record that is not already cached, can take longer. I've found that the potential downside of increased latency for roundtrips to be far outweighed by the robust caching.

Unbound caching

On the modern internet due to a variety of reasons, TTLs are often very short. Usually to the tune of minutes or seconds. This doesn't do much to help a small recursive resolver that's run locally. There is a slight fix for that, which is even commonplace with many ISPs.

Unbound has the ability to utilize a system by which records with expired TTLs stay in the cache and are served to clients. While simultaneously doing a new lookup and refreshing the cache with the latest record. Along with the above, Unbound can also prefetch popular queries that are about to expire.

```
serve-expired: <yes or no>
        If enabled, unbound attempts to serve old re-
        sponses from cache with a TTL of 0 in the re-
        sponse without waiting for the actual resolu-
        tion to finish. The actual resolution answer
        ends up in the cache later on. Default is
        "no".
```

Unbound installation

```
$ sudo apt install unbound
```

Once we have that downloaded, we also need to download a copy of the *root.hints* file. Remember this provides our resolver with the ability to begin resolution by communicating with one of the 13 root servers.

```
$ wget -O root.hints
https://www.internic.net/domain/named.root
```

```
$ sudo mv root.hints /var/lib/unbound/
```

The root.hints file does change but very infrequently and without much variation content-wise. It's good practice to re-download this file once or twice a year to make sure you have the most up to date info, when it comes to reaching the root servers.

Configs

The config file we want to create is:
/etc/unbound/unbound.d.confg/pi-hole.conf

Within here we can set options surrounding the interface and port, DNSSEC options, as well the location of our root hints file.

```
server:
    # If no logfile is specified, syslog is used
    # logfile: "/var/log/unbound/unbound.log"
    verbosity: 0

    interface: 127.0.0.1
    port: 5335
    do-ip4: yes
    do-udp: yes
    do-tcp: yes

    # May be set to yes if you have IPv6 connectivity
    do-ip6: no

    # You want to leave this to no unless you have *native* IPv6. With 6to4 and
    # Terredo tunnels your web browser should favor IPv4 for the same reasons
    prefer-ip6: no

    # Use this only when you downloaded the list of primary root servers!
    root-hints: "/var/lib/unbound/root.hints"

    # Trust glue only if it is within the server's authority
    harden-glue: yes
```

Config tweaks

You can also potentially tweak the cache size, bypass TTLs to a value you'd like and more. Read the *unbound.conf* man page for more options.

```
serve-expired: <yes or no>
       If enabled, unbound attempts to serve old responses from cache with a TTL of 0 in the response  without  waiting
       for  the  actual  resolution to finish.  The actual resolution answer ends up in the cache later on.  Default is
       "no".

serve-expired-ttl: <seconds>
       Limit serving of expired responses to configured seconds after expiration. 0 disables  the  limit.  This  option
       only applies when serve-expired is enabled. The default is 0.

serve-expired-ttl-reset: <yes or no>
       Set the TTL of expired records to the serve-expired-ttl value after a failed attempt to retrieve the record from
       upstream. This makes sure that the expired records will be served as long as there are queries for  it.  Default
       is "no".
```

Always exercise caution in not keeping expired records around for too long. It can delay propagation. When used properly, it's an efficient way to boost a recursive resolvers performance on a small network.

Starting & Testing

We can now start the service and test it's resolution.

```
$ sudo service unbound start
```

This can be followed up with the *dig* command to do a DNS lookup and verify our resolver is working.

```
$ dig pi-hole.net @127.0.0.1 -p 5335
```

```
; <<>> DiG 9.11.5-P4-5.1-Raspbian <<>> pi-hole.net @127.0.0.1 -p 5335
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18879
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
```

```
;; ANSWER SECTION:
pi-hole.net.            3600    IN      A       192.124.249.118

;; Query time: 288 msec
;; SERVER: 127.0.0.1#5335(127.0.0.1)
;; WHEN: Tue May 19 23:44:24 EDT 2020
;; MSG SIZE  rcvd: 56
```

DNSSEC Testing

Our recursive resolver is... resolving! Now lets test if DNSSEC is validating.

```
$ dig sigfail.verteiltesysteme.net @127.0.0.1 -p 5335
```

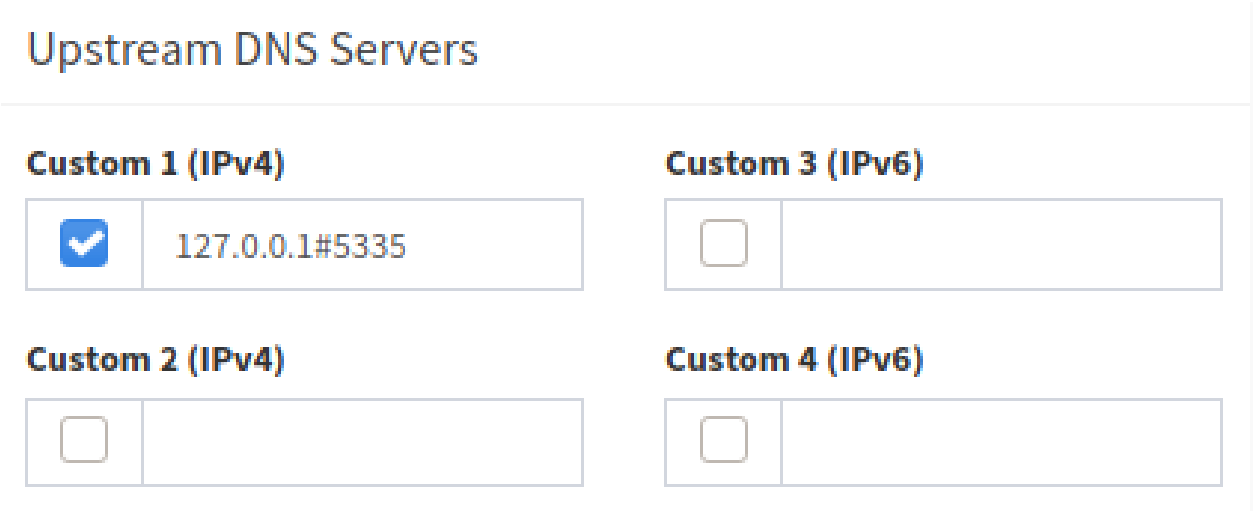
```
; <<>> DiG 9.11.5-P4-5.1-Raspbian <<>> sigfail.verteiltesysteme.net @127.0.0.1 -p 5335
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 6804
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1
```

```
$ dig sigok.verteiltesysteme.net @127.0.0.1 -p 5335
```

```
; <<>> DiG 9.11.5-P4-5.1-Raspbian <<>> sigok.verteiltesysteme.net @127.0.0.1 -p 5335
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 12762
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
```

Adding upstream

Now that our new Unbound resolver is validating and resolving queries, let's add it into Pi-hole so the entire network can begin to utilize it as their upstream DNS server.



You now have an open source one-stop shop for network-wide ad blocking, efficient caching and validation of recursive queries, along with potentially more control over your privacy and security!

Pi-hole limitations

No one solution is going to do it all. I covered the limitations of "ad blockers" in the beginning so let's talk about some regarding the Pi-hole.

- Hardcoded DNS

Many IoT devices come with DNS that is locked and can't be overridden. This is usually the case with smart speakers or streaming devices like Roku, etc. This means that there requests wont be going through your Pi-hole.

A solution exists if your router is capable of setting rules. Forcing all port 53 traffic on your network to your Pi-hole should help.

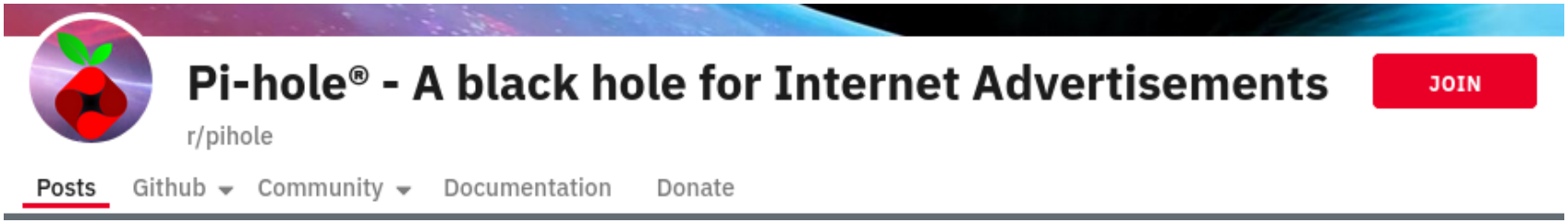
Pi-hole limitations

Some sites self-host their own ads or continuously serve ads from unique sub-domains. This makes them difficult to block as their main content and ads are being served on the same domain or constantly being rotated. YouTube and Spotify are just a few examples of where Pi-hole may not be fully utilized.

> Each domain starts with an `r` followed by a number. This number doesn't seem to go beyond 20. It is then followed by either three dashes, its unique ID (*fingerprint*) then `.googlevideo.com`

> e.g. `r7---sn-vgqs7ne7.googlevideo.com`

Pi-hole Questions



Should you have further questions or need help with specific issues, the Pi-hole Docs are a great source of info. Along with that you can check out their official forum, where you'll find a ton of up-to-date info regarding releases, new blocklists, etc.

- FAQs
- r/pihole

Layering

It's important to note that Pi-hole is only one layer of your ad blocking abilities on a network. It's still useful to have in-browser extensions for the cases above and more.

- Privacy Badger, ad and tracker blocking by the EFF.
- uBlock, wide spectrum content blocker.
- NoScript, lets you choose what javascript, java, flash and others run on the pages you visit.

Always make sure to do some research surrounding extensions and choose wisely!

Donate!

If you find that the open source or free software you use, makes your life easier. Consider donating to the developers!

Pi-hole donation

Unbound donation

Electronic Frontier Foundation

Slide deck

This slide deck was formatted/converted from Markdown using Beamer in Pandoc!