

# DNS

- [Apple Private Relay on pihole](#)
- [Blocking External Client DNS Queries](#)
- [Pi-Hole](#)
- [Redirecting Client DNS Requests](#)
- [Configure Conditional Forwarder on PiHole](#)
- [Redirect to a different domain - Cloudflared](#)
- [Redirect from root to WWW - cloudflared](#)
- [Redirect Domain to New Domain](#)

# Apple Private Relay on pihole

if your Apple device has a DNS issue, but the DNS queries are not showing in the pihole logs you should see something along the lines of `mask.icloud.com` and `mask-h2.icloud.com` being blocked as Blocked (Special Domain) nxdomain. This appears to be a problem with Apple Private Relay, which can happen even when this is disabled. Below are the steps to resolve the issue.

Open the pihole server and edit `/etc/pihole/pihole-FTL.conf` in a text editor of your choice

add the line `BLOCK_ICLOUD_PR=false`

Save the file and reboot the hardware

After reboot Apple device DNS queries should begin to show properly in the pihole, and the PR `mask.icloud.com` and `mask-h2.icloud.com` domains should no longer be visible.

# Blocking External Client DNS Queries

# Blocking External Client DNS Queries

This procedure configures the firewall to block DNS requests from local clients to servers outside the local network. With no other accessible DNS servers, clients are forced to send DNS requests to the DNS Resolver or DNS Forwarder on pfSense® software for resolution.

## Note

Blocking is effective but does not gracefully handle the situation. Clients must manually adjust their configuration to use the firewall for DNS. Redirecting DNS requests to the firewall is a more seamless solution. See [Redirecting Client DNS Requests](#) for details.

- Navigate to **Firewall > Rules, LAN** tab
- Create the block rule as the first rule in the list:
  - Click [fa level up](#) and or **Add to create** a new rule at the top of the list
  - Fill in the following fields on the rule:
    - Action
      - Reject*
    - Interface
      - LAN*
    - Protocol
      - TCP/UDP*
    - Destination
      - Any*
    - Destination Port Range
      - DNS (53)*
    - Description
      -
- Create the pass rule to allow DNS to the firewall, above the block rule:
  - Click [fa level up](#) and or **Add to create** a new rule at the top of the list
  - Fill in the following fields on the rule:

Action  
    *Pass*  
Interface  
    *LAN*  
Protocol  
    *TCP/UDP*  
Destination  
    *LAN Address*  
Destination Port Range  
    *DNS (53)*  
Description  
    Pass DNS to the Firewall

- Click **Apply Changes** to reload the ruleset

When complete, there will be two rule entries that look like the following picture:

 /images/blockdns.png

Certain local PCs could be allowed to use other DNS servers by placing a pass rule for them above the block rule.

## DNS over TLS

Another concern is that clients could use DNS over TLS to resolve hosts. DNS over TLS sends DNS requests over an encrypted channel on an alternate port, 853.

This traffic can be blocked with a firewall rule for port 853 using the same procedure used for 53. Though if the firewall will not be providing DNS over TLS service to clients, do not add the pass rule.

## DNS over HTTPS

Similar to DNS over TLS, clients may also use DNS over HTTPS (DoH). This is harder to block as it uses port 443. Blocking port 443 on common public DNS servers may help (e.g. 1.1.1.1, 8.8.8.8).

Some browsers automatically attempt to use DNS over HTTPS because they believe it to be more secure and better for privacy, though that is not always the case. Each browser may have its own methods of disabling this feature. Firefox uses a “canary” domain `use-application-dns.net` by default. If Firefox cannot resolve this name, Firefox disables DNS over HTTPS.

To prevent Firefox from using DNS over HTTPS, add the following to the DNS Resolver custom options:

```
server:  
local-zone: "use-application-dns.net" always_nxdomain
```

# Pi-Hole

Pi-Hole is a DNS server that has built in ability to block queries. It does this by returning 0.0.0.0 for queries on the block list.

[Pi-Hole Home Page](#)

For [installation instructions](#)

## List of Commands

Change password: `sudo pihole -a -p`  
Pi-Hole v6 updated command to

```
sudo pihole setpassword
```

Update: `pihole -up`

# Redirecting Client DNS Requests

# Redirecting Client DNS Requests

**Before you begin: Network level DNS must be set to use the pfSense firewall or DNS queries will fail. Attempting to redirect all DNS queries to your own DNS server, only to try and then send them off to Google or Cloudflared will fail.**

To restrict client DNS to only the DNS Resolver or Forwarder on pfSense® software, use a port forward to capture all client DNS requests.

Note

Either The DNS Resolver or DNS Forwarder must be active and it must bind to and answer queries on *Localhost*, or *All* interfaces.

See also

- [Blocking External Client DNS Queries](#)
- [Blocking Web Sites Using DNS](#)

The following example uses the LAN interface but the same technique will work with any local interface.

- Navigate to **Firewall > NAT, Port Forward** tab
- Click **fa level up** **Add** to create a new rule
- Fill in the following fields on the port forward rule:
  - Interface  
*LAN*
  - Protocol  
*TCP/UDP*
  - Destination  
**Invert Match** *checked, LAN Address*

Destination Port Range

*DNS (53)*

Redirect Target IP

127.0.0.1

Redirect Target Port

*DNS (53)*

Description

Redirect DNS

NAT Reflection

*Disable*

When complete, the port forward must appear as follows:

The image shows a port forward rule configuration. The rule name is 'Redirect DNS'. The destination port range is 'DNS (53)'. The redirect target IP is '127.0.0.1'. The redirect target port is 'DNS (53)'. The description is 'Redirect DNS'. The NAT reflection is set to 'Disable'.

#### Note

If DNS requests to other DNS servers are blocked, such as by following [Blocking External Client DNS Queries](#), ensure the rule to pass DNS to 127.0.0.1 is above any rule that blocks DNS.

With this port forward in place, DNS requests from local clients to **any** external IP address will result in the query being answered by the firewall itself. Access to other DNS servers on port 53 is impossible.

#### Tip

This can be adapted to allow access to only a specific set of DNS servers by changing the Destination network from “LAN Address” to an alias containing the allowed DNS servers. The **Invert match** box should remain checked.

#### Warning

Clients using DNS over TLS or DNS over HTTPS could circumvent this protection. Redirecting or blocking port 853 may help with DNS over TLS, depending on the clients.

See [Blocking External Client DNS Queries](#) for additional advice.

# Configure Conditional Forwarder on PiHole

## Conditional forwarding

If not configured as your DHCP server, Pi-hole typically won't be able to determine the names of devices on your local network. As a result, tables such as Top Clients will only show IP addresses.

One solution for this is to configure Pi-hole to forward these requests to your DHCP server (most likely your router), but only for devices on your home network. To configure this we will need to know the IP address of your DHCP server and which addresses belong to your local network. Exemplary input is given below as placeholder in the text boxes (if empty).

If your local network spans 192.168.0.1 - 192.168.0.255, then you will have to input `192.168.0.0/24`. If your local network is 192.168.47.1 - 192.168.47.255, it will be `192.168.47.0/24` and similar. If your network is larger, the CIDR has to be different, for instance a range of 10.8.0.1 - 10.8.255.255 results in `10.8.0.0/16`, whereas an even wider network of 10.0.0.1 - 10.255.255.255 results in `10.0.0.0/8`. Setting up IPv6 ranges is exactly similar to setting up IPv4 here and fully supported. Feel free to reach out to us on our [Discourse forum](#) in case you need any assistance setting up local host name resolution for your particular system.

You can also specify a local domain name (like `fritz.box`) to ensure queries to devices ending in your local domain name will not leave your network, however, this is optional. The local domain name must match the domain name specified in your DHCP server for this to work. You can likely find it within the DHCP settings.

Enabling Conditional Forwarding will also forward all hostnames (i.e., non-FQDNs) to the router when "Never forward non-FQDNs" is *not* enabled.

The following list contains all reverse servers you want to add. The expected format is one server per line in form of `<enabled>,<ip-address>[/<prefix-len>],<server>[#<port>][,<domain>]`. A valid config line could look like `true,192.168.0.0/24,192.168.0.1,fritz.box`

```
true,10.0.1.0/24,10.0.1.1,coltscomputer.services
```

Enter one entry per line like the above

# Redirect to a different domain - Cloudflared

[https://drive.google.com/file/d/1-uxqWIPed4vYTOqfIfMYyf\\_fkyS-vm8D/view?usp=drive\\_link](https://drive.google.com/file/d/1-uxqWIPed4vYTOqfIfMYyf_fkyS-vm8D/view?usp=drive_link)

The screenshot shows the Cloudflare dashboard interface. On the left is a navigation sidebar with categories like Analytics & Logs, DNS, Email, SSL/TLS, Security, Access, Speed, Caching, Workers Routes, Rules (highlighted), Network, Traffic, Custom Pages, Apps, Scrape Shield, Zaraz, and Web3. The main content area is titled 'testcastillo.com' and shows the configuration for an 'Edit Single Redirect' rule. The rule name is 'Redirect to a Different Domain [Template]'. Under 'If incoming requests match...', the 'Wildcard pattern' option is selected. The 'Request URL' field contains 'http://\*.testcastillo.com/\*'. Under 'Then...', the 'Target URL' field contains 'https://www.pumpalarm.com/' and the 'Status code' dropdown is set to '301'. A 'Go to...' search bar is visible in the top right corner.

**Cloudflare** Go to...

testcastillo.com Active Star Free plan

### Edit Single Redirect

Rule name (required)  
Redirect to a Different Domain [Template]  
Give your rule a descriptive name.

**If incoming requests match...**

- Wildcard pattern**  
Only apply the rule to requests matching the wildcard pattern
- Custom filter expression**  
Only apply the rule to requests matching the custom filter expression
- All incoming requests**  
Apply the rule to all requests

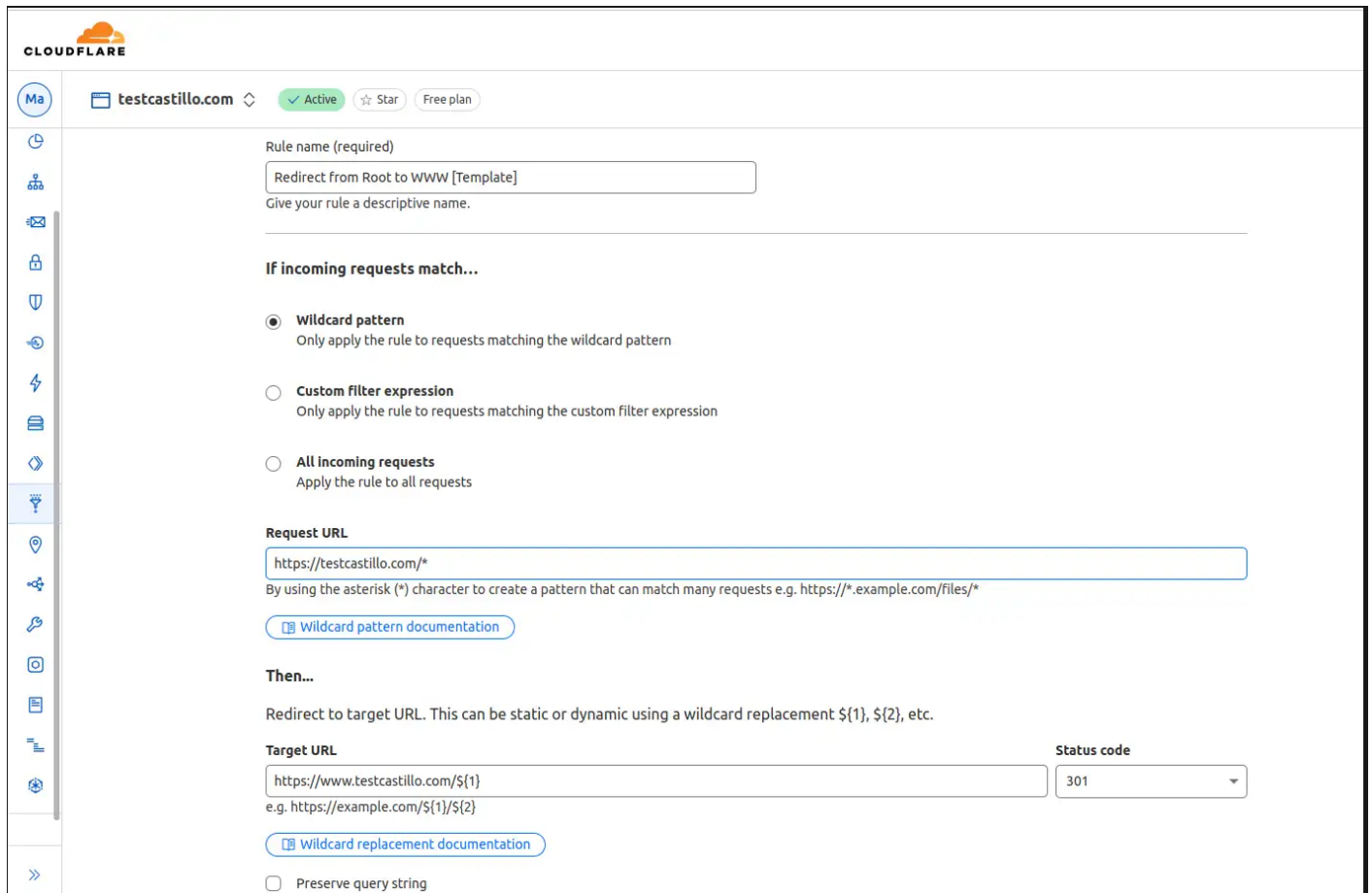
**Request URL**  
http://\*.testcastillo.com/\*  
By using the asterisk (\*) character to create a pattern that can match many requests e.g. https://\*.example.com/files/\*  
[Wildcard pattern documentation](#)

**Then...**  
Redirect to target URL. This can be static or dynamic using a wildcard replacement \${1}, \${2}, etc.

**Target URL** **Status code**  
https://www.pumpalarm.com/ 301  
e.g. https://example.com/\${1}/\${2}

# Redirect from root to WWW - cloudflared

[https://drive.google.com/file/d/1igk01QjlggVZXxe2pbHMNy\\_MtNCZoxkC/view?usp=drive\\_link](https://drive.google.com/file/d/1igk01QjlggVZXxe2pbHMNy_MtNCZoxkC/view?usp=drive_link)



**CLOUDFLARE**

testcastillo.com Active Star Free plan

Rule name (required)  
Redirect from Root to WWW [Template]  
Give your rule a descriptive name.

**If incoming requests match...**

- Wildcard pattern**  
Only apply the rule to requests matching the wildcard pattern
- Custom filter expression**  
Only apply the rule to requests matching the custom filter expression
- All incoming requests**  
Apply the rule to all requests

**Request URL**  
https://testcastillo.com/\*  
By using the asterisk (\*) character to create a pattern that can match many requests e.g. https://\*.example.com/files/\*  
[Wildcard pattern documentation](#)

**Then...**  
Redirect to target URL. This can be static or dynamic using a wildcard replacement \${1}, \${2}, etc.

**Target URL**  
https://www.testcastillo.com/\${1}  
e.g. https://example.com/\${1}/\${2}  
[Wildcard replacement documentation](#)

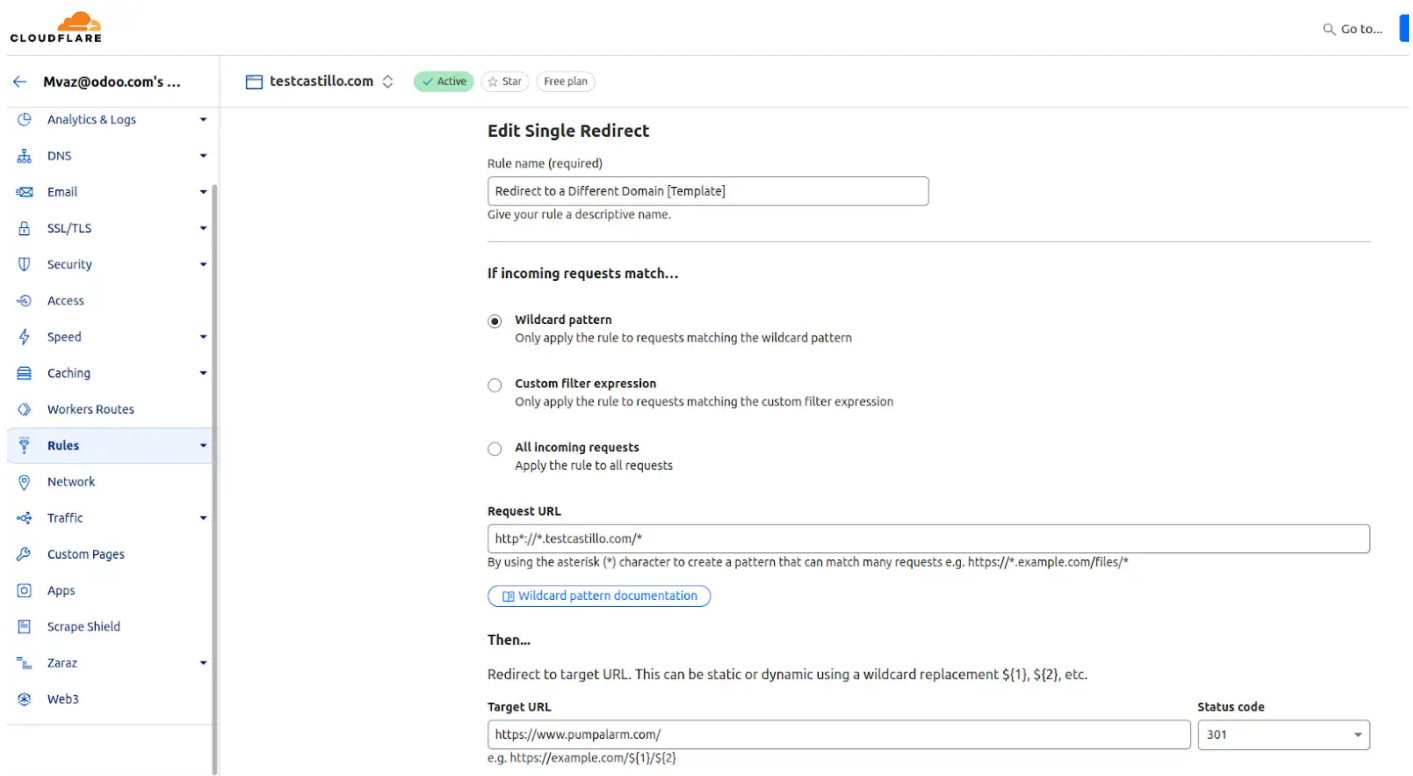
**Status code**  
301

Preserve query string

# Redirect Domain to New Domain

This config should allow you to redirect source.example.com to destination.example.com using cloudflared and Odoo as examples

First setup HTTP 301 redirect rules from the source to the destination domain.



The screenshot shows the Cloudflare dashboard interface for a domain named 'testcastillo.com'. The left sidebar contains a navigation menu with categories like Analytics & Logs, DNS, Email, SSL/TLS, Security, Access, Speed, Caching, Workers Routes, Rules (highlighted), Network, Traffic, Custom Pages, Apps, Scrape Shield, Zaraz, and Web3. The main content area is titled 'Edit Single Redirect' and contains the following configuration fields:

- Rule name (required):** A text input field containing 'Redirect to a Different Domain [Template]'. Below it is a note: 'Give your rule a descriptive name.'
- If incoming requests match...:** Three radio button options:
  - Wildcard pattern**: Only apply the rule to requests matching the wildcard pattern.
  - Custom filter expression**: Only apply the rule to requests matching the custom filter expression.
  - All incoming requests**: Apply the rule to all requests.
- Request URL:** A text input field containing 'http://\*.testcastillo.com/'. Below it is a note: 'By using the asterisk (\*) character to create a pattern that can match many requests e.g. https://\*.example.com/files/'. A link for 'Wildcard pattern documentation' is provided.
- Then...:** A section header followed by the instruction: 'Redirect to target URL. This can be static or dynamic using a wildcard replacement \${1}, \${2}, etc.'
- Target URL:** A text input field containing 'https://www.pumpalarm.com/'. Below it is a note: 'e.g. https://example.com/\${1}/\${2}'.
- Status code:** A dropdown menu set to '301'.

**CLOUDFLARE**

testcastillo.com Active Star Free plan

Rule name (required)  
  
 Give your rule a descriptive name.

**If incoming requests match...**

- Wildcard pattern**  
Only apply the rule to requests matching the wildcard pattern
- Custom filter expression**  
Only apply the rule to requests matching the custom filter expression
- All incoming requests**  
Apply the rule to all requests

**Request URL**  
  
 By using the asterisk (\*) character to create a pattern that can match many requests e.g. https://\*.example.com/files/\*  
[Wildcard pattern documentation](#)

**Then...**  
 Redirect to target URL. This can be static or dynamic using a wildcard replacement \${1}, \${2}, etc.

**Target URL**  
  
 e.g. https://example.com/\${1}/\${2}  
[Wildcard replacement documentation](#)

**Status code**

Preserve query string

Next configure the DNS records

Mvaz@odoo.com's ... testcastillo.com Active Star Free plan

**DNS Records**  
 Configure DNS records and review [proxy status](#) of your hostnames.  
[DNS documentation](#)

**Recommended steps to complete zone set-up** [Hide](#)

- ✓ Add an MX record for your **root domain** so that mail can reach @testcastillo.com addresses or [set up restrictive SPF, DKIM, and DMARC records](#) to prevent email spoofing. [New Alert](#)

**DNS management for testcastillo.com**  
 Review, add, and edit DNS records. Edits will go into effect once saved. DNS Setup: Full [Import and Export](#) [Dashboard Display Settings](#)

Search DNS Records  
  [Add record](#)

<input type="checkbox"/>	Type	Name	Content	Proxy status	TTL	Actions
<input type="checkbox"/>	A	testcastillo.com	34.173.148.21	Proxied	Auto	<a href="#">Edit</a>
<input type="checkbox"/>	CNAME	www	omnisite.odoo.com	Proxied	Auto	<a href="#">Edit</a>

**Cloudflare Nameservers**  
 Every DNS zone on Cloudflare is assigned a set of Cloudflare-branded nameservers.

This should redirect [www.source.example.com](http://www.source.example.com) and [source.example.com](http://source.example.com) to [www.destination.example.com](http://www.destination.example.com).

#Cloudflared #Odoo #DNS #Redirect