

Using Cloudflare

Cloudflare DNS:

If you wish to use Cloudflare for DNS ONLY, you can configure it just like any other DNS provider, and simply [follow the steps for adding a custom domain as usual](#).

WARNING: Your domain DNS entry **must have a GRAY cloud, not an ORANGE (proxied) cloud next to the entry.**



Using Cloudflare's proxy for your domain without additional configuration will cause all incoming connections to fail!



This is the correct image shown for DNS-only entries.

Cloudflare Proxy

Even though Elest.io automatically provides SSL and has a firewall, there can be advantages to using Cloudflare for Proxying traffic, notably DDoS attacks and automatic filtering of scripted attacks.

Note: Cloudflare only proxies traffic on certain ports. If you want to use this hostname for SSH, FTP, or other services whose ports are not listed in the above link, you must configure Cloudflare to provide DNS only or use Cloudflare's Spectrum offer.

Because Elest.io already creates an SSL certificate for your website trusted by a root CA, the recommended configuration is to set Cloudflare to use Strict SSL verification when connecting to your server.

Before continuing, ensure you have already configured the domains [as per the instructions on the previous page](#).

Option 1: To set up strict SSL verification for your **whole domain**:

1. Navigate to the `SSL/TLS` section of your domain's dashboard.
2. Select the `"Full (strict)"` option.
3. Your changes will be saved automatically. You're done!

The screenshot shows the Cloudflare SSL/TLS dashboard. On the left is a navigation menu with categories like Overview, Security, Access, Speed, Caching, Workers Routes, Rules, Network, Traffic, and Apps. The main content area displays a confirmation message: "Your SSL/TLS encryption mode is Full (strict)" with a sub-note "This setting was last changed 2 hours ago". Below this is a diagram showing the flow from a Browser to Cloudflare and then to an Origin Server, with lock icons indicating encryption. To the right of the diagram are four radio button options: "Off (not secure)", "Flexible", "Full", and "Full (strict)". The "Full (strict)" option is selected. Below the options is a link to "Learn more about End-to-end encryption with Cloudflare." and a button to "Create a Configuration Rule to customize these settings by hostname." At the bottom right of the dashboard are links for "API" and "Help".

Option 2: To set up strict SSL verification for a **specific subdomain**:

1. In your domain's dashboard, navigate to `Rules > Configuration Rules` and click `Create Rule`
2. Name your rule, and configure the incoming request filters.

Edit Configuration Rule

Rule name (required)

Give your rule a descriptive name.

If...

When incoming requests match...

- All incoming requests
The rule will apply to all traffic
- Custom filter expression
The rule will only apply to traffic matching the custom expression

When incoming requests match...

Field	Operator	Value	
<input type="text" value="Hostname"/>	<input type="text" value="equals"/>	<input type="text" value="subdomain.example.com"/> e.g. example.com	<input type="button" value="And"/> <input type="button" value="Or"/>

Expression Preview

[Edit expression](#)

```
(http.host eq "subdomain.example.com")
```

3. Configure the SSL to

SSL (optional)

Configure SSL settings available in SSL/TLS > Overview tab.

Select SSL/TLS encryption mode

Cancel

Save

4. Click

Option 3: Manual configuration (Advanced)

If you need a custom implementation, you can disable the creation of an SSL certificate with the following steps.

Create a CNAME record for your Cloudflare entry and point to the CNAME provided for that service in the Elestio dashboard.

These changes can be overwritten in the future if you modify the list of domains via the Elest.io dashboard.

1) Connect to the VM with SSH and type this:

```
nano /opt/elestio/nginx/.env
```

there remove your domain from the first line and save with CTRL+X

then type this command:

```
cd /opt/elestio/nginx;  
docker-compose down;  
docker-compose up -d;
```

After that, nginx won't try again to obtain an SSL certificate for your domain.

Revision #4

Created 17 July 2023 15:44:13 by Daniel

Updated 17 July 2023 16:35:17 by Daniel