HOSTING CTF WITH CTFd 2.1.4

Written by: ArkAngels

Note: In this documentation the writer assumes the reader knows how to set the domain needed to proceed. This documentation was based on the writer's experience and some of them might lack a thing or two. The documentation may be updated over time.

The writer will explain how to host CTFd with **docker-compose** and reverse proxy **nginx**.

Setting up CTFd

First, install docker, docker-compose, nginx, and git.

- https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-16-04 -> docker
- https://www.digitalocean.com/community/tutorials/how-to-install-docker-compose-on-ubuntu-16-04 -> docker-compose
- https://www.digitalocean.com/community/tutorials/how-to-install-nginx-on-ubuntu-16-04 -> nginx
- **[sudo] apt install git**

After that, clone the CTFd repository. Use the latest version.

- https://github.com/CTFd/CTFd -> CTFd -> clone it

In the dockerized version of CTFd, the compose file will create 3 containers:

- CTFd
- MariaDB
- Redis (Cache server)

Try to build the docker using the following command:

```
[sudo] docker-compose up
```

If there’s an error like this:

<todo>

Just remove the line

```
Internal: true
```

Inside the docker-compose.yml file.

If there’s no error like the above, then you’re good.

Usually this error occurs at a non fresh installed VPS

And then if there is no other error, you can simply run it in the background.

```
[sudo] docker-compose up -d
```

On this version, the CTFd should be running at http://0.0.0.0:8000. If you’re happy with this setting, then you’ve reached the end of this guide.

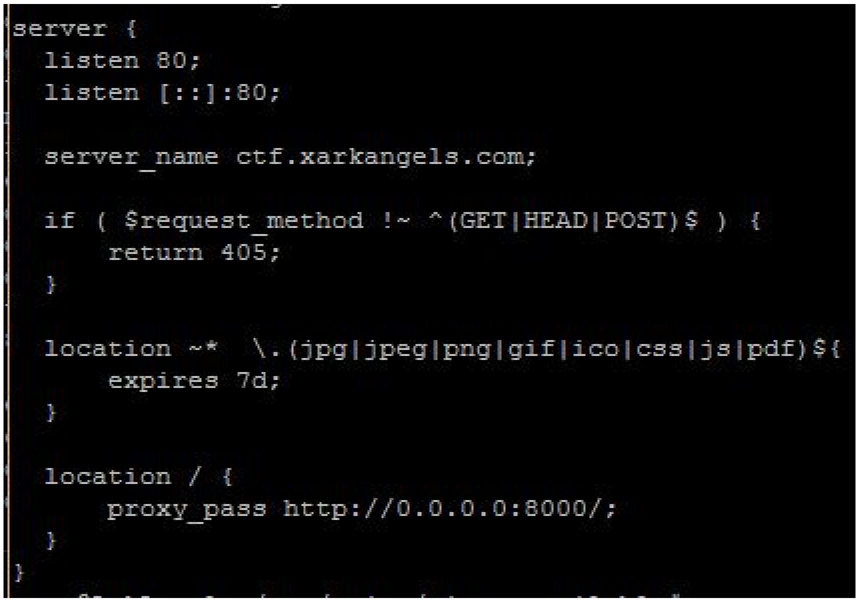
Use Nginx Reverse Proxy

If you decide to use a reverse proxy so that you don’t have to write the port on the URL, in this documentation, the writer will show you how to set a reverse proxy with Nginx.

First, install nginx:

- **[sudo] apt install nginx**

After that, go to **/etc/nginx/conf.d/** and create a config file **[your domain].conf.** In that config file, simply create the reverse proxy setting. The writer's looks like this:



```
server {
  listen 80;
  listen [::]:80;

  server_name ctf.xarkangels.com;

  if ( $request_method !~ ^(GET|HEAD|POST)$ ) {
      return 405;
  }

  location ~*  \.(jpg|jpeg|png|gif|ico|css|js|pdf)${
      expires 7d;
  }

  location / {
      proxy_pass http://0.0.0.0:8000/;
  }
}
```

After that, simply copy the config file to **/etc/nginx/sites-available** and create the symlink to **/etc/nginx/sites-enabled.**

[sudo] ln -s /etc/nginx/sites-available/[yourdomain].conf /etc/nginx/sites-enabled/[yourdomain].conf

And that's it! Now the reverse proxy should work (except if you haven't configure your dns yet).

WARNING!

If you want to use Cloudflare **DO NOT** combine it with Certbot SSL Certificate Redirect Rule as it will cause a problem (Redirect too many times).

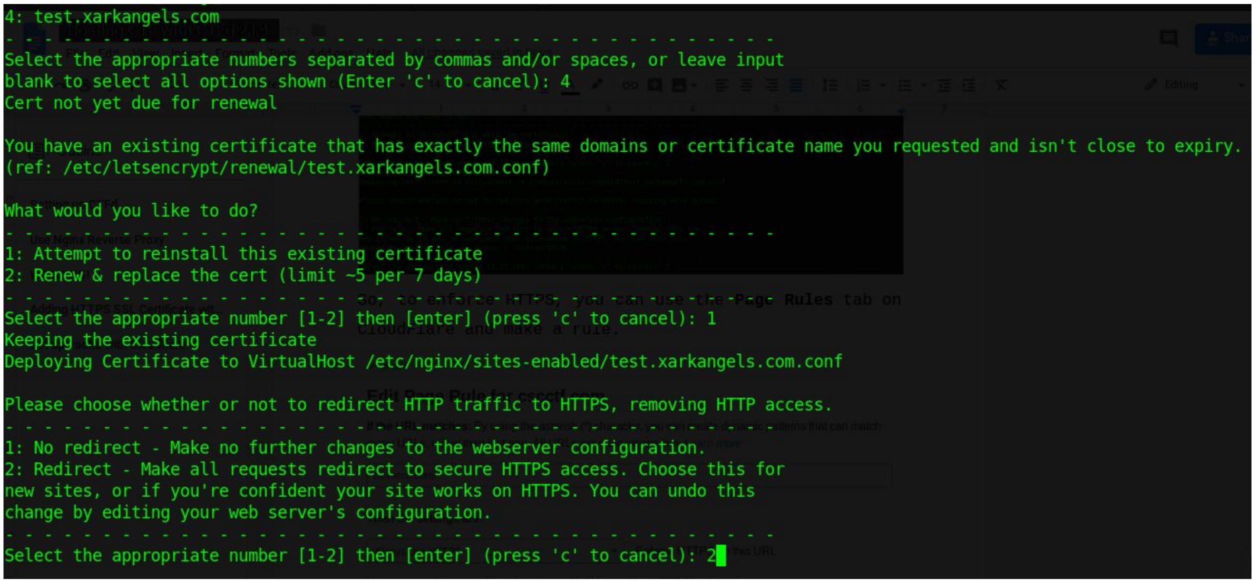
Adding HTTPS SSL Certificate with Certbot

There are 2 ways of installing certbot:

- From apt which has a few steps:
  - **[sudo] apt-get update**
  - **[sudo] apt-get install software-properties-common**
  - **[sudo] add-apt-repository ppa:certbot/certbot**
  - **[sudo] apt-get update**
  - **[sudo] apt-get install certbot certbot-nginx**
- From python pip:
  - **[sudo] python -m pip install certbot certbot-nginx**

In this documentation, the writer will show you how to set certbot from python.

After you install the certbot, simply run **certbot** and choose to set HTTPS on all websites and then you’re done!



```
4: test.xarkangels.com
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Select the appropriate numbers separated by commas and/or spaces, or leave input
blank to select all options shown (Enter 'c' to cancel): 4
Cert not yet due for renewal

You have an existing certificate that has exactly the same domains or certificate name you requested and isn't close to expiry.
(ref: /etc/letsencrypt/renewal/test.xarkangels.com.conf)

What would you like to do?
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
1: Attempt to reinstall this existing certificate
2: Renew & replace the cert (limit ~5 per 7 days)
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Select the appropriate number [1-2] then [enter] (press 'c' to cancel): 1
Keeping the existing certificate
Deploying Certificate to VirtualHost /etc/nginx/sites-enabled/test.xarkangels.com.conf

Please choose whether or not to redirect HTTP traffic to HTTPS, removing HTTP access.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
1: No redirect - Make no further changes to the webserver configuration.
2: Redirect - Make all requests redirect to secure HTTPS access. Choose this for
new sites, or if you're confident your site works on HTTPS. You can undo this
change by editing your web server's configuration.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Select the appropriate number [1-2] then [enter] (press 'c' to cancel): 2
```

Set up cache with Cloudflare

Now if you want to secure your website from irresponsible third party who like to attack your website, you can use Cloudflare to secure your website.

First, after you get your CloudFlare account, you can add your domain to CloudFlare.

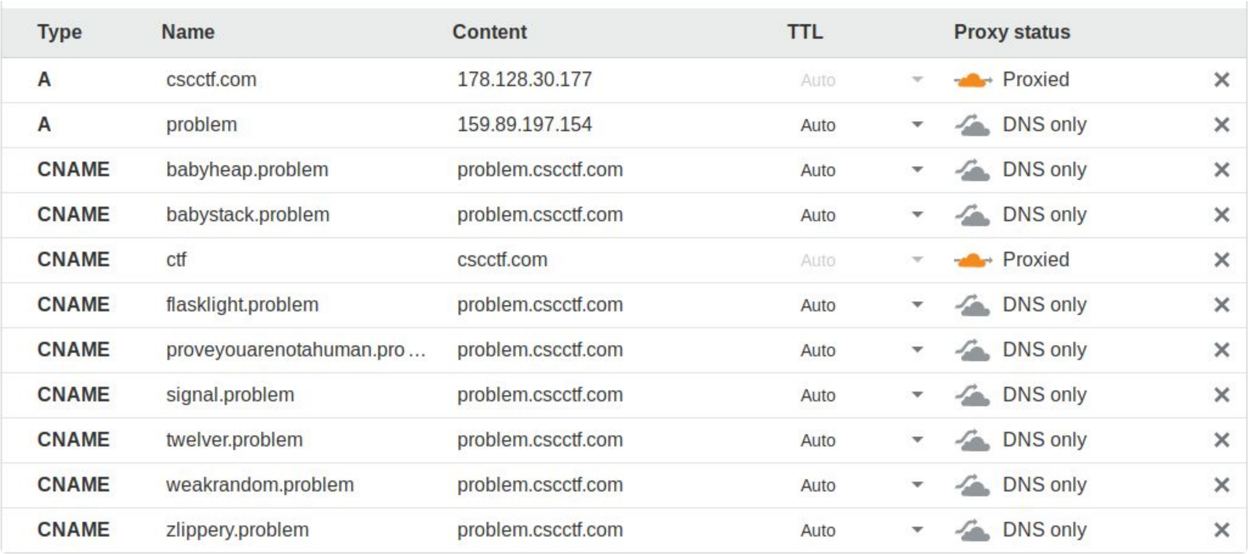
Second, select the free plan (Except if you have more budgets for a paid plan, but the writer don’t think that’s needed for hosting a simple CTF).

Third, check your DNS (this can also be done later).

Fourth, change your NameServer (NS) to the one provided with cloudflare, **BUT** remember to change it in your domain page not your VPS page. If it’s not clear enough:

- The writer was using DigitalOcean for the writer’s VPS and the writer bought the domain from GoDaddy. If the writer want to use CloudFlare, the writer need to change those NameServers from ns1.digitalocean.com, ns2.digitalocean.com, and ns3.digitalocean.com to the one provided by CloudFlare from the manage DNS Page in GoDaddy which in the writer’s case those were dawn.ns.cloudflare.com and theo.ns.cloudflare.com.

After that, you can start by managing your DNS.



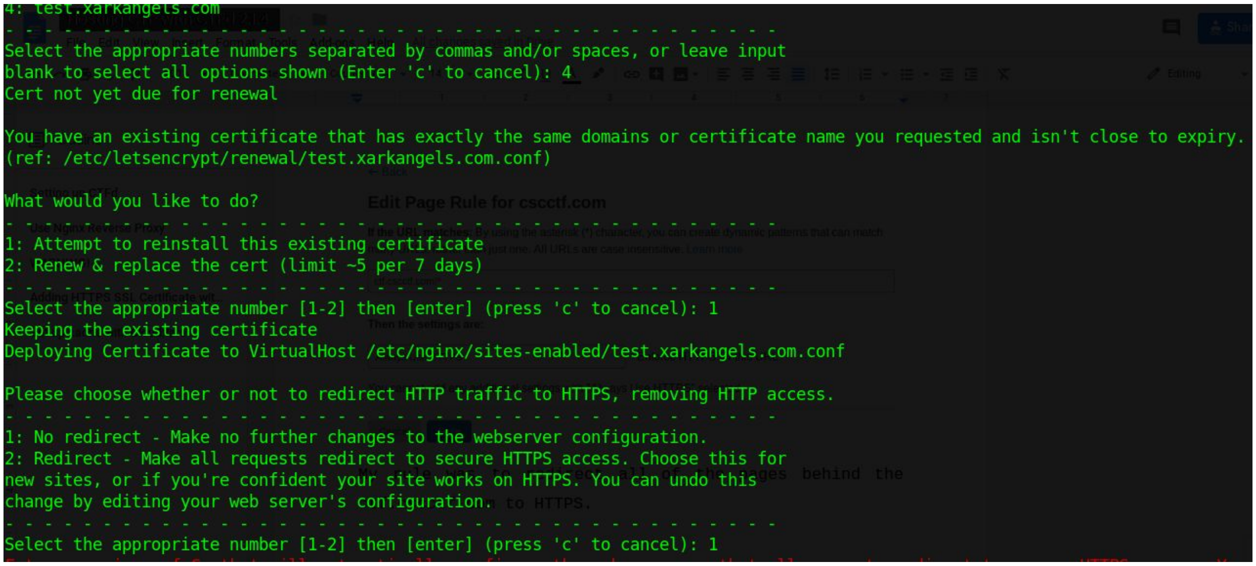
Type	Name	Content	TTL	Proxy status	
A	cscctf.com	178.128.30.177	Auto	Proxied	✕
A	problem	159.89.197.154	Auto	DNS only	✕
CNAME	babyheap.problem	problem.cscctf.com	Auto	DNS only	✕
CNAME	babystack.problem	problem.cscctf.com	Auto	DNS only	✕
CNAME	ctf	cscctf.com	Auto	Proxied	✕
CNAME	flasklight.problem	problem.cscctf.com	Auto	DNS only	✕
CNAME	proveyouarenotahuman.pro …	problem.cscctf.com	Auto	DNS only	✕
CNAME	signal.problem	problem.cscctf.com	Auto	DNS only	✕
CNAME	twelver.problem	problem.cscctf.com	Auto	DNS only	✕
CNAME	weakrandom.problem	problem.cscctf.com	Auto	DNS only	✕
CNAME	zlippery.problem	problem.cscctf.com	Auto	DNS only	✕

As you can see, there are records that the writer set to DNS Only and Proxied. Most of these are because the writer don’t want to screw the IP resolution if the writer set it to Proxied. And we **NEED** to set the DNS Record on **BOTH** DigitalOcean Manage DNS Page **AND** on the CloudFlare DNS Page or otherwise it won’t be accessible. So the writer decided to just set the main record like the platform one to proxied.

Also, the writer used 2 droplets for this CTF:

- One to specifically handle the CTFd Platform.
- One where we hosted all of the problems.

As for the certificate, remember to use Certbot first to obtain the certificate, and remember that when you want to use CloudFlare, you need to set the Certbot to **No Redirect** as if you do, it will cause an error (Too many redirects).



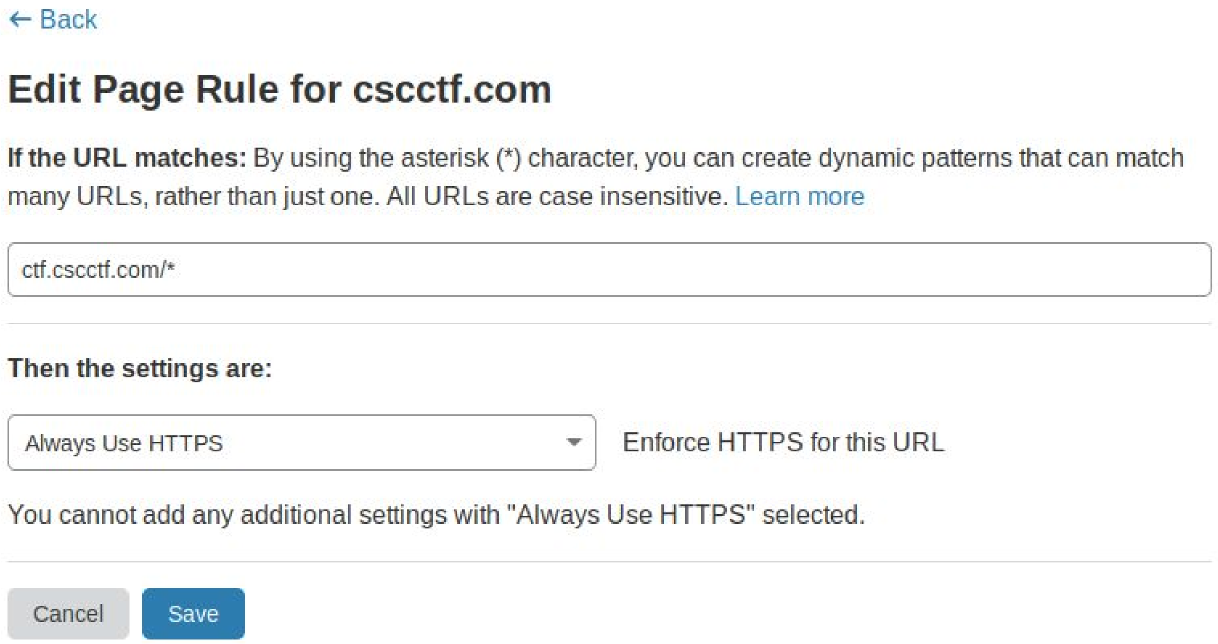
```
4: test.xarkangels.com
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Select the appropriate numbers separated by commas and/or spaces, or leave input
blank to select all options shown (Enter 'c' to cancel): 4
Cert not yet due for renewal

You have an existing certificate that has exactly the same domains or certificate name you requested and isn't close to expiry.
(ref: /etc/letsencrypt/renewal/test.xarkangels.com.conf)

What would you like to do?
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
1: Attempt to reinstall this existing certificate
2: Renew & replace the cert (limit ~5 per 7 days)
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Select the appropriate number [1-2] then [enter] (press 'c' to cancel): 1
Keeping the existing certificate
Deploying Certificate to VirtualHost /etc/nginx/sites-enabled/test.xarkangels.com.conf

Please choose whether or not to redirect HTTP traffic to HTTPS, removing HTTP access.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
1: No redirect - Make no further changes to the webserver configuration.
2: Redirect - Make all requests redirect to secure HTTPS access. Choose this for
new sites, or if you're confident your site works on HTTPS. You can undo this
change by editing your web server's configuration.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Select the appropriate number [1-2] then [enter] (press 'c' to cancel): 1
```

So, to enforce HTTPS, you can use the **Page Rules** tab on CloudFlare and make a rule.



Edit Page Rule for cscctf.com

If the URL matches: By using the asterisk (*) character, you can create dynamic patterns that can match many URLs, rather than just one. All URLs are case insensitive. Learn more

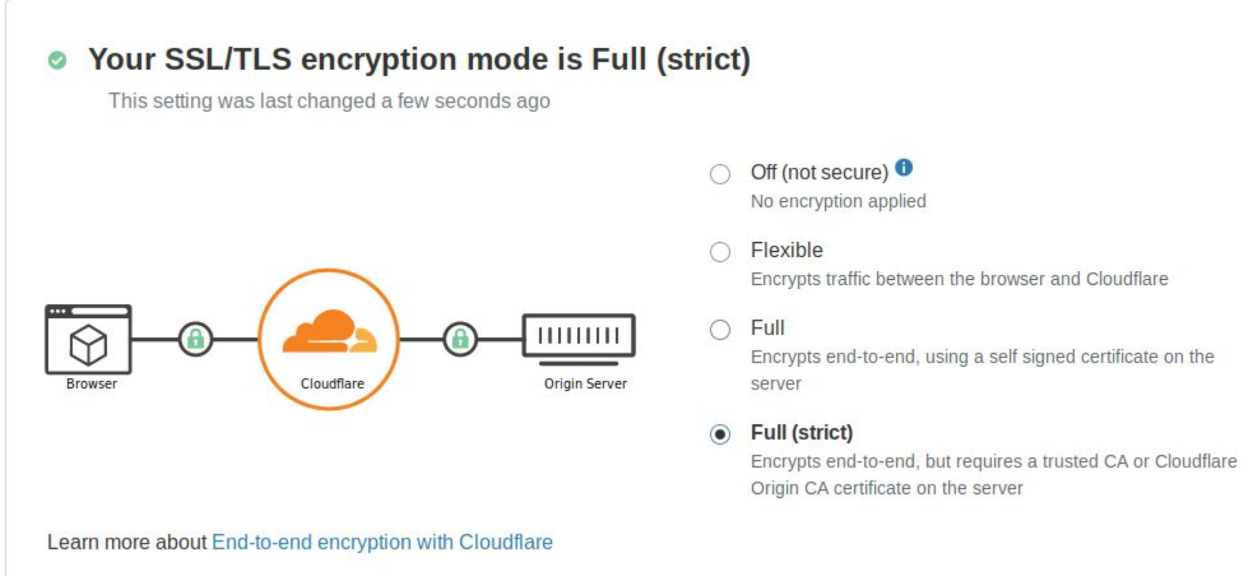
ctf.cscctf.com/*

Then the settings are:

Always Use HTTPS — Enforce HTTPS for this URL

You cannot add any additional settings with "Always Use HTTPS" selected.

The writer rule was to redirect all of the pages behind the ctf.cscctf.com to HTTPS. If you still concern about the HTTPS Redirect, you can change the rule in SSL tab in CloudFlare to Full or Full(Strict). The writer used Full(Strict) here.



Your SSL/TLS encryption mode is Full (strict)

This setting was last changed a few seconds ago

○ Off (not secure)
No encryption applied

○ Flexible
Encrypts traffic between the browser and Cloudflare

○ Full
Encrypts end-to-end, using a self signed certificate on the server

● **Full (strict)**
Encrypts end-to-end, but requires a trusted CA or Cloudflare Origin CA certificate on the server

Learn more about End-to-end encryption with Cloudflare

And after this you should be fine.