

Configuring Traefik for the dns-01 challenge with OVH as DNS provider

Encrypt all the things!



Antoine Hamon [Follow](#)

Feb 27, 2019 · 7 min read



Since you shouldn't have landed on this page without knowing what Traefik, Let's Encrypt and OVH are, I won't extend myself introducing them 😊

As you should know Let's Encrypt is able to deliver x509 certificates for your domains after having validated your website/domain identity. This consists in resolving one of the four challenge types: http-01, tls-sni-01 (soon outdated), tls-alpn-01 or dns-01 (introduced early 2018). While most administrators uses the http or tls challenge (easier to configure), I will be writing here about the dns-01 one as it offers more possibilities (such as delivering wildcard

certificates) and as it is, according to me, the most secured one.

You can actually find some other tutorials on the Internet dealing with the same subject, but I wasn't really satisfied with them as they give Traefik too much access rights to the OVH API... That was not something suitable to me (they usually give all rights under the '/domain' path, or worse under the '/' path! 🤯).

Why should you consider using the dns-01 challenge?

Well I know that using the dns-01 challenge might be impossible in a lot of companies for security concerns as it requires to give rights to Traefik to create and remove some DNS records (TXT entries), but on the other hand it permits you to have wildcard certificates (I'm not really a huge fan of them), you will be able to have certificate for internal services (that are not facing the Internet/resolves to a private IP), and more especially it will prevent your domains and certificates to be listed on some specialized websites such as <https://crt.sh> (example with the medium.com domain 🤯).

Configuring OVH

To allow Traefik to create and remove DNS records, you first need to create an application account for Traefik to interact with the OVH API. To generate a new application account, go to the web page listed below, and follow instructions:

Europe: <https://eu.api.ovh.com/createApp/>

Canada/USA: <https://ca.api.ovh.com/createApp/>

Creating an API Application

Please fill the following fields to create an API application.

Account ID or email address	<input type="text" value="my-ovh-account"/>
Password	<input type="password" value="Password"/>
Application name	<input type="text" value="traefik-letsencrypt"/>
Application description	<input type="text" value="My Traefik proxy / Let's Encrypt"/>
<input type="button" value="Create keys"/>	

Creating an API Application

Application created

Your application credentials:

Application Name	<input type="text" value="traefik-letsencrypt"/>
Application Description	<input type="text" value="My Traefik proxy / Let's Encrypt"/>
Application Key	<input type="text" value="ya9xB3cE6mR1Zkhn"/>
Application Secret	<input type="text" value="WdrqTkATVNkhVGtLaCzFWIbNclbMimz6"/>

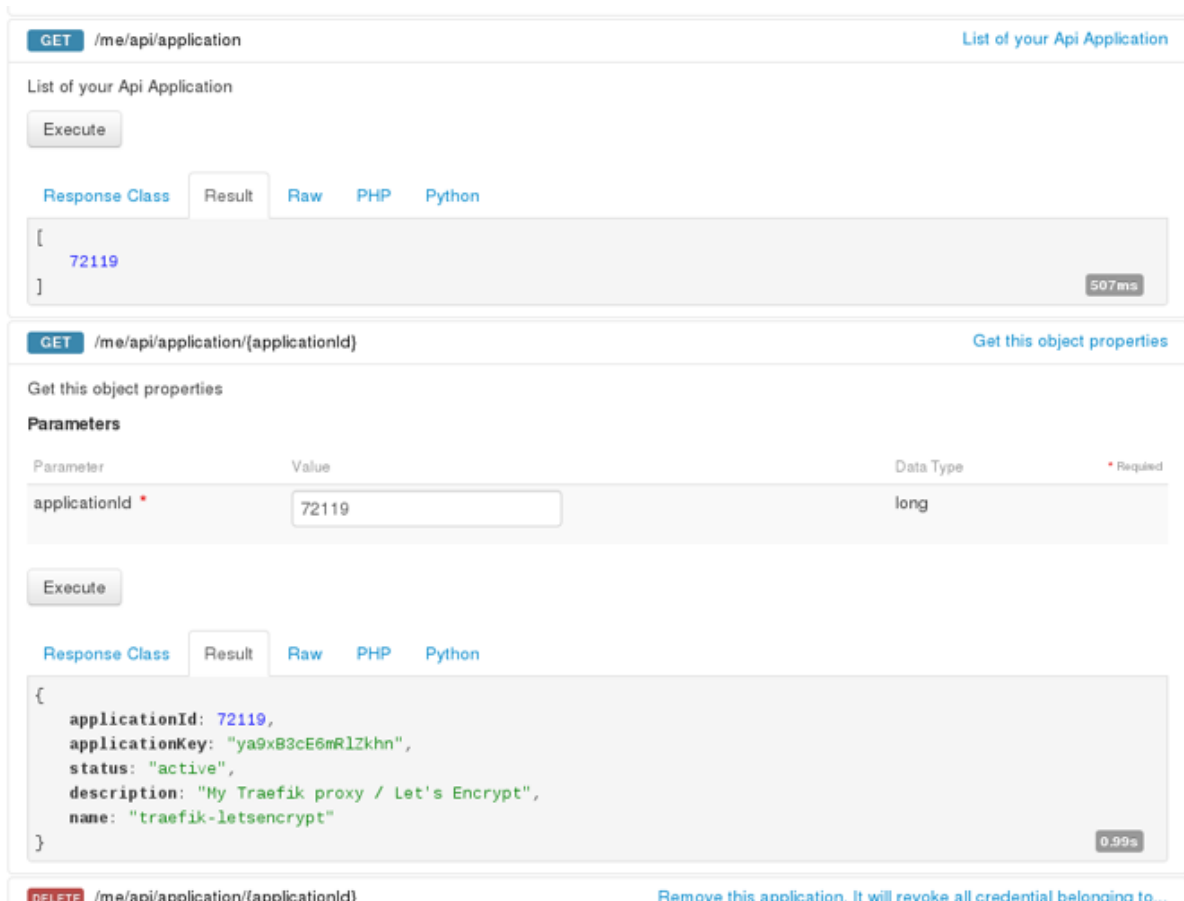
You don't really need to write down nor save those credentials as we will be using them very soon (and you will be able to find them later on your Traefik configuration). Furthermore, if you lost them you can always delete and re-create them (see at the

end of this chapter for the deletion).

Now that your application account has been created, you can check its existence in the OVH API console:

Europe: <https://eu.api.ovh.com/console>

Canada/USA: <https://ca.api.ovh.com/console>



GET /me/api/application List of your Api Application

List of your Api Application

Execute

Response Class Result Raw PHP Python

```
[
  {
    72119
  }
]
```

507ms

GET /me/api/application/{applicationId} Get this object properties

Get this object properties

Parameters

Parameter	Value	Data Type	* Required
applicationId *	72119	long	

Execute

Response Class Result Raw PHP Python

```
{
  applicationId: 72119,
  applicationKey: "ya9xB3cE6mR1Zkhn",
  status: "active",
  description: "My Traefik proxy / Let's Encrypt",
  name: "traefik-letsencrypt"
}
```

0.99s

DELETE /me/api/application/{applicationId} Remove this application. It will revoke all credential belonging to...

Yeah! We have it listed 😊

Next we need to create some access rules for this application account. To resolve the dns-01 challenge Traefik should be able to create a TXT DNS record, refresh the zone and delete the record. We can check which calls are performed by Traefik looking at the source code of the acme library it's using (xenolf/lego).

As giving access right to an application account can't be done from the OVH API console, we need to do it from a terminal with a curl command (or any other program that can perform HTTP requests). The request looks as follow:

```
curl -XPOST -H "X-Ovh-Application: <my_application_key>"  
-H "Content-type: application/json"  
https://eu.api.ovh.com/1.0/auth/credential -d '{  
  "accessRules": [  
    {"method": "POST", "path": "/domain/zone/<my_domain>/record"  
  },  
  
    {"method": "POST", "path": "/domain/zone/<my_domain>/refresh"  
  },  
  
    {"method": "DELETE", "path": "/domain/zone/<my_domain>/record/*"}  
  ],  
  "redirection": "https://www.<my_domain>"  
}'
```

The 'redirection' parameter tell which web page the application should be redirected to once logged in. This is quite useless in our case but it still must be filled.

If you want Traefik to manage multiple domains, you may either duplicate those access rules for each domain, or replace '<my_domain>' with '' which will affect all domains.*

Also, OVH sadly do not permit us restraining the entry type to TXT



So with this tutorial data the request looks as following:

```
curl -XPOST -H "X-Ovh-Application: ya9xB3cE6mRlZkhn" -H
```

```
"Content-type: application/json"
https://eu.api.ovh.com/1.0/auth/credential -d
'{"accessRules":
[{"method":"POST","path":"/domain/zone/example.com/record"},
{"method":"POST","path":"/domain/zone/example.com/refresh"},
{"method":"DELETE","path":"/domain/zone/example.com/record/*"}], "redirection": "https://www.example.com"}'
```

If the request process correctly, it should return a JSON with following informations:

```
{"validationUrl":"https://eu.api.ovh.com/auth/?
credentialToken=zPIM6Qyln7q2pwzCxU26RzVvXCZo7QHXAjXMYNrRl
jt1EXdEq7Uk5sKNSJNiNw82","state":"pendingValidation","con
sumerKey":"JlJ64Fwtc2yost2WjuIzxBXNiEE0QJ6C"}
```

The 'consumerKey' will later be used in Traefik configuration.

Access rights have been created and attached to the application account but it still needs validation (notice the state as being 'pendingValidation'). You have to click on the validation URL, and enter **your** credentials to validate the process (you should also double-check that the application name and access rights listed on the web page are correct). Remember to select the validity period to 'Unlimited' as we don't want to repeat the procedure regularly.



Permission to access your account

Please enter your password to allow the application to access your account.

Account ID or email address

Password

Validity

Application name: traefik-letsencrypt

Application description: My Traefik proxy / Let's Encrypt

Access Granted :

- POST on /domain/zone/example.com/record
- POST on /domain/zone/example.com/refresh
- DELETE on /domain/zone/example.com/record/*

After validating, you will be redirected to : <https://www.example.com>

After hitting the “Log in” button if you get redirected to the web page you filed on your request everything went fine. You can still verify that rules have been attached to your application account by checking on the OVH API console:

[Delete](#) [Remove this application. It will remove all credential belonging to...](#)

[GET /me/api/credential](#) [List of your Api Credentials](#)

List of your Api Credentials

Parameters

Parameter	Value		Data Type	* Required
applicationId	<input type="text" value="72119"/>	Filter the value of applicationId property (like)	long	
status	<input type="text"/>	Filter the value of status property (=)	auth.CredentialStateEnum	

[Response Class](#) [Result](#) [Raw](#) [PHP](#) [Python](#)

```
[  
  471496378  
]
```

222ms

[GET /me/api/credential/{credentialId}](#) [Get this object properties](#)

Get this object properties

Parameters

Parameter	Value	Data Type	* Required
credentialId *	<input type="text" value="471496378"/>	long	

Response Class Result Raw PHP Python

```
{
  applicationId: 72119,
  - rules: [
    - {
      method: "POST",
      path: "/domain/zone/example.com/record"
    },
    - {
      path: "/domain/zone/example.com/refresh",
      method: "POST"
    },
    - {
      method: "DELETE",
      path: "/domain/zone/example.com/record/*"
    }
  ],
  ovhSupport: false,
  status: "validated",
  expiration: null,
  creation: "2019-02-27T15:49:36+01:00",
  credentialId: 471496378,
  lastUse: null
}
```

173ms

DELETE /me/api/credential/{credentialId} [Remove this credential](#)

If this is the case you are now all set on the OVH side! 

Deleting OVH account for Traefik

If one day you think your OVH application account have been compromised or you just want to delete it, this also can be done from the OVH API console:

GET /me/api/application/{applicationId} [Get this object properties](#)

Get this object properties

Parameters

Parameter	Value	Data Type	* Required
applicationId *	<input type="text" value="72119"/>	long	

Execute

Response Class Result Raw PHP Python

```
{
  applicationId: 72119,
  applicationKey: "ya9xB3cE6mRlZkhn",
  status: "active",
  description: "My Traefik proxy / Let's Encrypt",
  name: "traefik-letsencrypt"
}
```

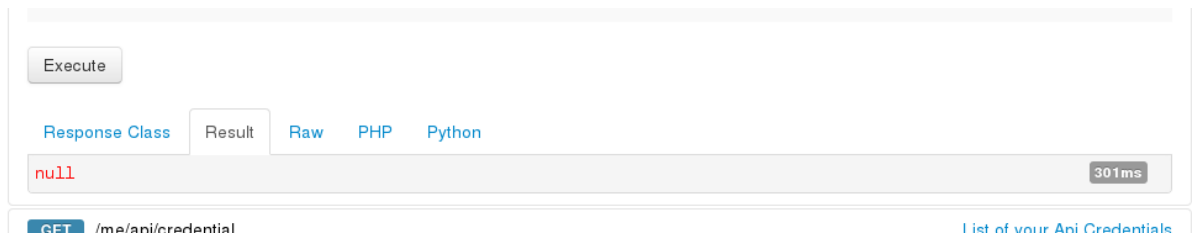
0.99s

DELETE /me/api/application/{applicationId} [Remove this application. It will revoke all credential belonging to...](#)

Remove this application. It will revoke all credential belonging to this application.

Parameters

Parameter	Value	Data Type	* Required
applicationId *	<input type="text" value="72119"/>	long	



(You can verify it has correctly been deleted by checking the list of applications).

. . .

Configuring Traefik

The Traefik part is way easier and straight forward compared to the OVH one.

Traefik configuration has to be done in two different places: in its main configuration file (traefik.toml/traefik.yaml) and providing OVH credential to the Traefik container.

Providing OVH credentials is common to every Traefik versions. Edit your docker-compose file (or any equivalent) and add the following variables to the environment section:

```
environment:
- OVH_ENDPOINT=ovh-eu (or ovh-ca)
- OVH_APPLICATION_KEY=<your_application_key>
- OVH_APPLICATION_SECRET=<your_application_secret>
- OVH_CONSUMER_KEY=<your_application_consumer_key>
```

Replacing those fields with data from this example:

```
environment:
- OVH_ENDPOINT=ovh-eu
- OVH_APPLICATION_KEY=ya9xB3cE6mRlZkhn
- OVH_APPLICATION_SECRET=WdrqTkATVNkhVGtLaCzFWIbNclbMimz6
- OVH_CONSUMER_KEY=JlJ64Fwtc2yost2WjuIzxBXNiEE0QJ6C
```

Traefik v2.x (v1.x below)

Traefik — Let's Encrypt documentation

As I find the YAML format more convenient I'm be using it for this tutorials, but paths are the same for TOML files.

Create a **certificatesResolvers** root section in your configuration file:

```
certificatesResolvers:
  myResolverName:
    acme:
      email: acme@example.com
      storage: /etc/traefik/acme.json
      dnsChallenge:
        provider: ovh
        delayBeforeCheck: 10
```

👉 ***Beware that if you are running multiple Traefik instances, they cannot share the same 'acme.json' file for concurrency reason. You will need to use the Enterprise Edition.***

The 'delayBeforeCheck' option tells Traefik how many seconds it should wait before verifying the DNS TXT entry (and then trigger Let's Encrypt challenge verification). I had some issue setting it to 0 so I set it to 10 (time delay isn't a real issue when it comes to certificate generation as they are renewed 30 days in advance).

You also have the possibility to generate wildcard domains certificates.

💡 You do not need to restart the Traefik container for it to take config changes into account if you activate the ‘watch’ option.

To finish, you just need to follow Traefik documentation to configure your Docker containers/Kubernetes ingress to create endpoints.

Traefik v1.x

Traefik — Let’s Encrypt documentation

In the traefik.toml file, under your ‘acme’ part you need to create a ‘acme.dnsChallenge’ section containing the following:

```
[acme]
email = "letsencrypt@example.com"
storage = "/etc/traefik/acme.json"
entryPoint = "https"
[acme.dnsChallenge]
provider = "ovh"
delayBeforeCheck = 10
```

👉 *Beware that if you are running multiple Traefik instances, they cannot share the same ‘acme.json’ file for concurrency reason. You will need to use a key-value store for your configuration.*

The ‘delayBeforeCheck’ option tells Traefik how many seconds it should wait before verifying the DNS TXT entry (and then trigger Let’s Encrypt challenge verification ; official

documentation). I had some issue setting it to 0 so I set it to 10 (time delay isn't a real issue when it comes to certificate generation as they are renewed few days in advance).

You also have the possibility to ask Let's Encrypt to generate wildcard domains certificates by adding a 'acme.domains' subsection (official documentation):

```
[[acme.domains]]  
main = "*.example.com"  
...
```

Restart the Traefik container and you're done with the config



To finish, you need to configure your Docker containers/Kubernetes ingresses to create endpoints.

. . .

If you want to read more on the subject:

A Technical Deep Dive: Securing the Automation of ACME DNS
Challenge Validation

)

Discover Medium

Welcome to a place where words matter. On Medium, smart voices and original ideas take center stage - with no ads in sight. Watch

Make Medium yours

Follow all the topics you care about, and we'll deliver the best stories for you to your homepage and inbox. Explore

Become a member

Get unlimited access to the best stories on Medium — and support writers while you're at it. Just \$5/month. Upgrade

[About](#)

[Help](#)

[Legal](#)