Certbot Wildcards Certifiactes (OVH)

Wildcard Let’s Encrypt Certificate with OVH DNS (Certbot)

0. Introduction

Goal

Obtain and automatically renew a wildcard TLS certificate (***.qool.ovh**) using Let’s Encrypt, with DNS hosted at OVH and the server hosted elsewhere (e.g. Contabo).

Key Requirements

Wildcard certificates require DNS-01 validation. This means Certbot must be able to create and remove DNS TXT records via the OVH API.

High-Level Overview

The process consists of:

1. Installing Certbot (Snap version)
2. Creating a correct OVH API token
3. Storing OVH credentials securely
4. Verifying API access with Python
5. Running Certbot with the OVH DNS plugin
6. Cleaning up and relying on auto-renewal
7. 1. Install Certbot (Snap)

1. Installing Certbot (Snap version)

Remove old packages

```
sudo apt remove certbot
```

Install Snap and Certbot

```
sudo apt update
sudo apt install snapd
sudo snap install core
sudo snap refresh core
sudo snap install --classic certbot
sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

Install OVH DNS plugin

```
sudo snap set certbot trust-plugin-with-root=ok
sudo snap install certbot-dns-ovh
```

2. Create OVH API Token

Where

Create the token at:

https://api.ovh.com/createToken/

Token settings

Validity: **Unlimited**

Required permissions (exact)

```
GET    /domain/zone
GET    /domain/zone/
GET    /domain/zone/qool.ovh
GET    /domain/zone/qool.ovh/*
POST   /domain/zone/qool.ovh/*
```

```
PUT    /domain/zone/qool.ovh/*
DELETE /domain/zone/qool.ovh/*
```

Note: OVH treats `/domain/zone` and `/domain/zone/` as different paths. Certbot (Lexicon) requires the trailing slash permission.

wise.ovh, these are the API settings

```
GET    /domain/zone/
GET    /domain/zone/*
POST   /domain/zone/*
PUT    /domain/zone/*
DELETE /domain/zone/*
```

When API key is set, this can be executed:

```
sudo certbot certonly \
  --dns-ovh \
  --dns-ovh-credentials ~/.secrets/certbot/ovh.ini \
  -d wise.ovh -d '*.wise.ovh'
```

3. Store OVH Credentials Securely

Create credentials file

```
sudo nano /etc/letsencrypt/ovh.ini
```

File contents

```
dns_ovh_endpoint = ovh-eu
dns_ovh_application_key = YOUR_APPLICATION_KEY
dns_ovh_application_secret = YOUR_APPLICATION_SECRET
dns_ovh_consumer_key = YOUR_CONSUMER_KEY
```

Lock down permissions

```
sudo chmod 600 /etc/letsencrypt/ovh.ini
```

4. Verify OVH API Access (Before Certbot)

Create a small Python test

```
import ovh

client = ovh.Client(
    endpoint='ovh-eu',
    application_key='YOUR_APPLICATION_KEY',
    application_secret='YOUR_APPLICATION_SECRET',
    consumer_key='YOUR_CONSUMER_KEY'
)

print(client.get('/domain/zone'))
```

Expected output

A list of domains including **qool.ovh**. If this works, the API token and permissions are correct.

5. Request the Wildcard Certificate

Run Certbot

```
sudo certbot certonly \
  --dns-ovh \
  --dns-ovh-credentials /etc/letsencrypt/ovh.ini \
  --dns-ovh-propagation-seconds 120 \
  --agree-tos \
  --email admin@qool.ovh \
  -d "*.qool.ovh" \
  -d "qool.ovh"
```

Successful result

Certbot reports that the certificate was issued and stored in:

```
/etc/letsencrypt/live/qool.ovh/
```

6. Verify and Test Renewal

Check certificate files

```
sudo ls -l /etc/letsencrypt/live/qool.ovh/
```

Test auto-renewal

```
sudo certbot renew --dry-run
```

Snap installs a systemd timer automatically, so renewals run without manual action.

7. Cleanup (Recommended)

Remove duplicate credential copies

Keep only:

```
/etc/letsencrypt/ovh.ini
```

Remove test artifacts

```
rm -f ~/python/set-ovy.py
rm -rf ~/ovh-test
```

Final Notes

- DNS hosting location matters; web hosting location does not.
- Certbot OVH plugin requires both `/domain/zone` and `/domain/zone/` permissions.
- Protect the OVH API token like a root password.
- Once working, the setup is fully automatic and production-safe.

Revision #7
Created 2026-02-09 20:33:33 UTC by Max
Updated 2026-07-06 19:55:20 UTC by Max