Certbot Let's Encrypt client

START HERE

`sudo certbot --nginx`
Get and install a cert

`systemctl list-timers`
Is auto-renew set up?

`sudo certbot certonly --webroot -w <dir> -d <domain>`
Get a cert only

`sudo certbot certonly --standalone -d <domain>`
Use a temporary server

`sudo certbot certificates`
List managed certs

`sudo certbot renew --dry-run`
Test renewal

`sudo certbot renew`
Renew due certs

`sudo certbot delete --cert-name <name>`
Remove a cert

Request certonly or run → `challenge` → **Validate** HTTP-01 or DNS-01 → `issue` → **Store** live/<name>/ → `renew` → **Renew** timer + hooks

CONCEPTS

Certbot needs root. Point your server at /etc/letsencrypt/live/<name>/, never at the archive copies.

Certs are renewed from their saved settings in renewal/<name>.conf, so the plugin you used first is the one renewal uses.

OBTAIN

`certbot` **`certonly -d <domain>`**
Get a cert, do not install it

`certbot` **`--nginx -d <domain>`**
Get a cert and edit nginx config

`-d <a> -d <b>`
Several names on one cert

`--cert-name <name>`
Name the cert (file paths, housekeeping)

`--expand -d <old> -d <new>`
Add a name to an existing cert
Give every name the cert should end up with, not only the new one.

OBTAIN (cont.)

`--staging`
Use the test CA (no rate-limit risk)

`--key-type rsa / ecdsa`
Pick the key algorithm
ecdsa is the default since Certbot 2.0.

AUTHENTICATORS

`--webroot -w <dir>`
Write challenge files into a live web root
Each -d uses the most recent -w, so put -w before its domains.

`--standalone`
Run a temporary server on port 80
Stop whatever listens on port 80 first, or use a hook.

`--dns-cloudflare / --dns-route53`
DNS-01 via a DNS provider plugin

`--dns-cloudflare-credentials <file>`
Credentials file for the DNS plugin
Keep it chmod 600.

INSPECT AND RENEW

`certbot` **`certificates`**
List certs, names, expiry and paths

`certbot` **`renew`**
Renew every cert that is due

`certbot` **`renew --dry-run`**
Test renewal against the staging CA

`certbot` **`renew --cert-name <name>`**
Renew one cert only

`--force-renewal`
Renew even if not due
Repeated forced renewals hit rate limits.

`--deploy-hook <cmd>`
Run after a successful renewal only

`--pre-hook <cmd>`
Run before each renewal attempt

`--post-hook <cmd>`
Run after each renewal attempt

`certbot` **`install --cert-name <name>`**
Deploy an existing cert to the server

MANAGE CERTS

`certbot` **`delete --cert-name <name>`**
Delete a cert and its renewal config
Remove it from the server config first, or the server will not start.

`certbot` **`revoke --cert-name <name>`**
Revoke a cert at the CA

FILES AND HOOKS

Under /etc/letsencrypt

Path	Holds
`live/<name>/fullchain.pem`	Cert plus chain; use this
`live/<name>/privkey.pem`	Private key
`live/<name>/cert.pem, chain.pem`	Parts, rarely needed
`archive/<name>/`	Numbered history
`renewal/<name>.conf`	Saved options for renew
`renewal-hooks/{pre,deploy,post}/`	Executable scripts
`cli.ini`	Global defaults

FILES AND HOOKS (cont.)

Hook scripts in renewal-hooks/ run in alphabetical order. Deploy hooks run only when a cert was actually renewed.

Logs are in /var/log/letsencrypt/. Never edit or delete files in /etc/letsencrypt by hand; use certbot delete.

Most packages install a systemd timer (certbot.timer) or cron job; check before adding your own.

RECIPES

Webroot cert, reload nginx on renew

```
sudo certbot certonly --webroot \
  -w /var/www/example \
  -d example.com -d www.example.com \
  --deploy-hook 'systemctl reload nginx'
```

RECIPES (cont.)

Wildcard via Cloudflare DNS

```
# /root/cf.ini holds dns_cloudflare_api_token = ...
chmod 600 /root/cf.ini
sudo certbot certonly --dns-cloudflare \
  --dns-cloudflare-credentials /root/cf.ini \
  -d example.com -d '*.example.com'
```

Standalone with the web server paused

```
sudo certbot renew \
  --pre-hook 'systemctl stop nginx' \
  --post-hook 'systemctl start nginx'
```

Auto-renew with cron if no timer exists

```
# /etc/cron.d/certbot
0 3,15 * * * root certbot renew -q \
  --deploy-hook 'systemctl reload nginx'
```