Getting Started

Anyone who has gone through the trouble of setting up a secure website knows what a hassle getting and maintaining a certificate can be. Let's Encrypt automates away the pain and lets site operators turn on and manage HTTPS with simple commands.

No validation emails, no complicated configuration editing, no expired certificates breaking your website. And of course, because Let's Encrypt provides certificates for free, no need to arrange payment.

This page describes how to carry out the most common certificate management functions using the Let's Encrypt client. You're welcome to use any compatible client, but we only provide instructions for using the client that we provide.

If you'd like to know more about how this works behind the scenes, check out our how it works page.

Installing Client Software

If your operating system includes a packaged copy of the `letsencrypt` client, install it from there and use the `letsencrypt` command. Otherwise, you can use our `letsencrypt-auto` wrapper script to get a copy quickly:

```
$ git clone https://github.com/letsencrypt/letsencrypt
$ cd letsencrypt
$ ./letsencrypt-auto --help
```

`letsencrypt-auto` accepts the same flags as `letsencrypt`; it installs all of its own dependencies and updates the client code automatically (but it's comparatively slow and large in order to achieve that).

Limits on usage

Let's Encrypt will issue a limited number of certificates each week. See this thread for the latest numbers. If you are trying out the client for the first time, you may want to use the `--test-cert` flag, and a domain name that does not receive live traffic. This will get certificates from our `staging` server. They won't be valid in browsers, but otherwise the process will be the same, so you can test a variety of configuration options without hitting the rate limit.

How To Use The Client

The Let's Encrypt client supports a number of different "plugins" that can be used to obtain and/or install certificates. A few examples of the options are included below:

If you're running Apache on a recent Debian-based OS, you can try the Apache plugin, which automates both obtaining and installing certs:

```
letsencrypt --apache
```

On other platforms automatic installation is not yet available, so you will have to use the `certonly` command. Here are some examples:

To obtain a cert using the "webroot" plugin, which can work with the webroot directory of any webserver software:

```
letsencrypt certonly --webroot -w /var/www/example -d example.com -d
www.example.com -w /var/www/thing -d thing.is -d m.thing.is
```

This command will obtain a single cert for example.com, www.example.com, thing.is, and m.thing.is; it will place files below /var/www/example to prove control of the first two domains, and under /var/www/thing for the second pair.

To obtain a cert using a built-in "standalone" webserver (you may need to temporarily stop your existing webserver, if any) for example.com and www.example.com:

```
letsencrypt certonly --standalone -d example.com -d www.example.com
```

Renewing a Certificate

As of version 0.4.0, the letsencrypt python client supports a high-level renew subcommand that attempts to renew all of your certs using the same settings that you originally used to obtain them. You can test it out by running:

```
letsencrypt renew --dry-run
```

which will obtain test certs, and shouldn't leave any persistent changes on your system. If you like the results, you can run:

```
letsencrypt renew
```

If you want to renew specific certificates (rather than all of them) or tweak the exact parameters used for renewal, you can use the `letsencrypt certonly` command to perform renewal of a single certificate with more specific control of settings. When using `letsencrypt certonly`, you can obtain renewal of a single certificate at a time. Specify `-d` flags for each and every domain covered by the domain that you wish to renew.

NOTE: As of 0.4.0, the client will record any settings you select when using `certonly` or `renew`, and future uses of renew will use the most recent ones. In version 0.3.0, the client would only record settings chosen the first time a cert was obtained, and not those provided when it was replaced or renewed.

The `renew` verb is designed for partly or fully unattended use, so it also implies `--non-interactive`. This flag means that the client will never stop to prompt you for the answer to a question; this is great for an automated renewal, but you should ensure all of your settings are correctly set, since you will not be able to specify options interactively while the renewal underway.

The `--dry-run` flag will fetch a certificate from our `staging` server. The obtained certificate isn't saved to disk and your configuration isn't updated, but it will allow you to test if your `renew` or `certonly` command will work for renewal. Certificates gotten with the staging server will not count against your rate limit on the production server.

If you want to change previously specified values, you can specify new options on the command line when renewing, for example:

```
letsencrypt renew --rsa-key-size 4096
```

Running `letsencrypt renew` will renew all certificates that are within the renewal window (by default, certificates within 30 days of expiry). If you would like to renew a single certificate, you should call `letsencrypt certonly -d` with the exact same domains that are specified in the certificate you are trying to renew. For example:

`letsencrypt certonly --keep-until-expiring --webroot -w /var/www/example.com -d example.com,www.example.com -w /var/www/thing -d thing.is,m.thing.is`

If your certificate is installed in a local server, you'll need to reload that server's configuration once the `certonly` command completes (for instance, `server apache2 reload` for an `apache2` web server).

Writing your own renewal script

We will be providing scripts to automate this renewal in the next few weeks, but if you wish to roll your own renewal script in the mean time, you can put a `renew` invocation of the client in a script. If your certificate is installed in a local server application, you may want to cause the script to reload the server's configuration after renewal completes. If you've used the `standalone` authentication method, you may want to cause the script to stop the webserver before the invocation of the Let's Encrypt client and restart the webserver afterward.

For instance, apache plugin users might want a script like this:

```
#!/bin/sh
if ! /path/to/letsencrypt-auto renew > /var/log/letsencrypt/renew.log
    echo Automated renewal failed:
    cat /var/log/letsencrypt/renew.log
    exit 1
fi
```

While users of the standalone plugin might want a script like this:

```
#!/bin/sh
service nginx stop  # or whatever your webserver is
/path/to/letsencrypt-auto renew -nvv --standalone > /var/log/letsencry
LE_STATUS=$?
service nginx start # or whatever your webserver is
if [ "$LE_STATUS" != 0 ]; then
```

```
    echo Automated renewal failed:
    cat /var/log/letsencrypt/renew.log
    exit 1
fi
```

You can test your script by adding a `--force-renewal` flag; that will force a renewal even if your cert isn't close to expiry. If you have an OS packaged version of the client, call `letsencrypt` rather than `/path/to/letsencrypt-auto`. Note that scripts which call `letsencrypt-auto` will auto-update the letsencrypt client, for better and worse. If you don't want that, but obtained the client with `letsencrypt-auto`, you can call the client directly inside the virtual environment it created: `/home/user/.local/share/letsencrypt/bin/letsencrypt` rather than `/path/to/letsencrypt-auto`.

Once you're happy with your script, you can run it with `cron` or `systemd`. We recommend running renewal scripts at least daily, at a random hour and minute. This gives the script many days to retry renewal in case of transient network failures or server outages.

Forthcoming releases will include a default script to be run from crontab or systemd , and customizable schedules for renewal.

Note: the instructions above assume that you have the latest version of the client, 0.4.0. If you are using packages of an older version, you may want to look at older instructions for renewal.

Revoking a Certificate

The following command can be used to revoke a particular certificate.

```
$ letsencrypt revoke --cert-path example-cert.pem
```

Full Documentation

For more information on the official client, please see the full documentation. Known issues are tracked on GitHub. Please check closely for known issues before filing new ones.

Getting Help

After reading the documentation and issues list, if you need additional help, please try our helpful community forums.

letsencrypt
letsencrypt

Let's Encrypt is a free, automated, and open certificate authority brought to you by the Internet Security Research Group (ISRG). ISRG is a California public benefit corporation, and is recognized by the IRS as a tax-exempt organization under Section 501(c)(3) of the Internal Revenue Code. ISRG's mission is to reduce financial, technological, and education barriers to secure communication over the Internet.

660 York Street, San Francisco, CA 94110