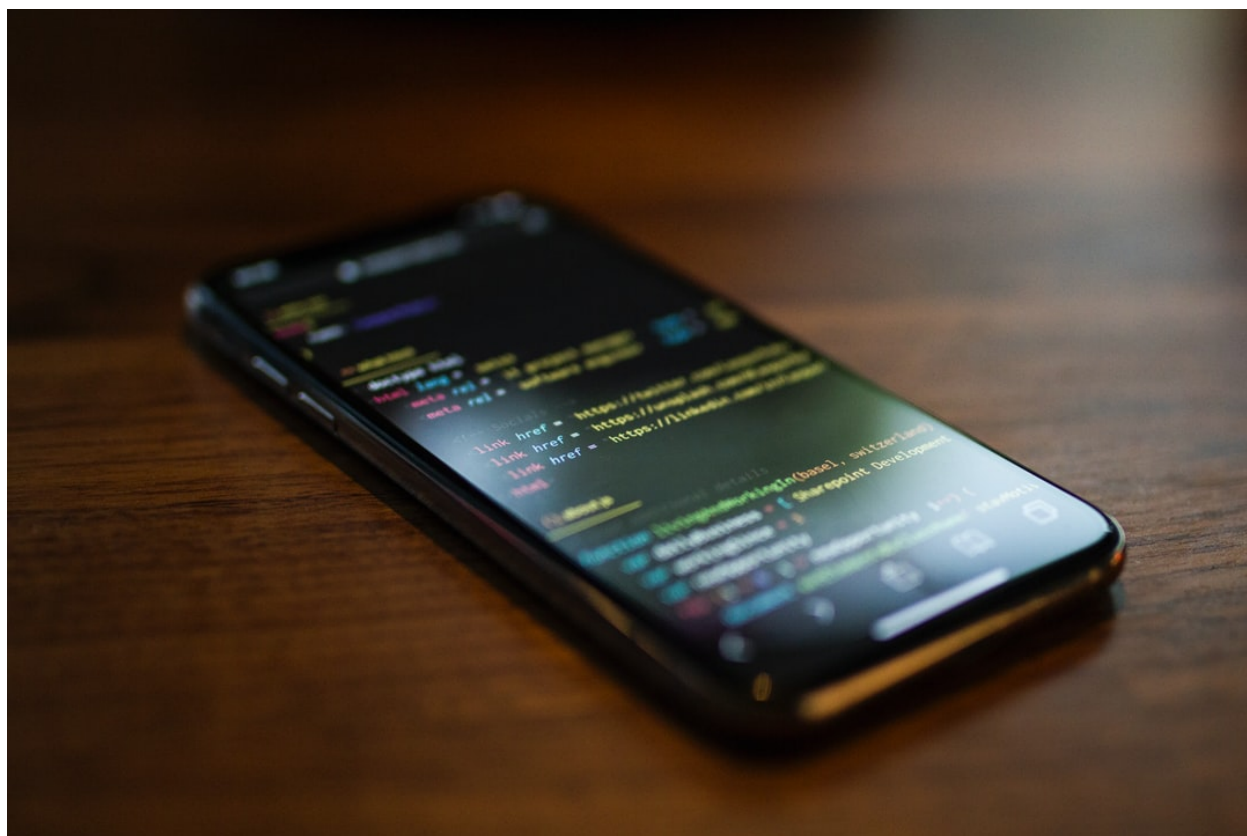


JANUARY 19, 2018 / #HTTPS

# How to get HTTPS working on your local development environment in 5 minutes



Daksh Shah



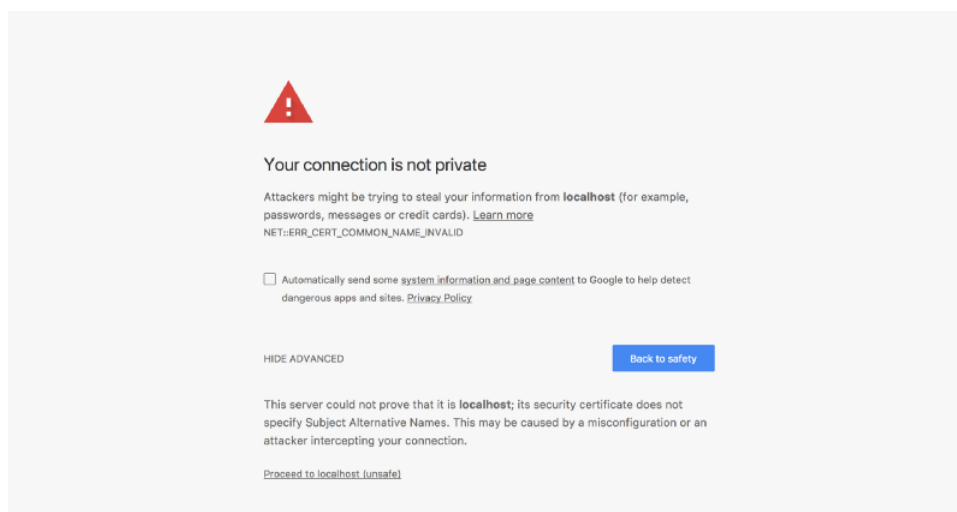
```
> Listening at https://localhost:443
```

Almost any website you visit today is protected by HTTPS. If yours isn't yet, it should be. Securing your server with HTTPS also means that you can't send requests to this server from one that isn't protected by HTTPS. This poses a problem for developers who use a local development environment because all of them run on `http://localhost` out-of-the-box.

Learn to code — free 3,000-hour curriculum

with HTTPS as part of a move to enhance security. I ran into a situation where my local development environment's requests to the server started getting rejected.

A quick Google search later, I found several articles like [this](#), [this](#) or [this one](#) with detailed instructions on how I could implement HTTPS on `localhost`. None of these instructions seemed to work even after I followed them religiously. Chrome always threw a `NET::ERR_CERT_COMMON_NAME_INVALID` error at me.



## The problem

All the detailed instructions I had found were correct for the time they were written. Not anymore.

After a ton of Googling, I discovered that the reason for my local certificate getting rejected was that [Chrome had deprecated support for commonName matching in certificates](#), in effect, requiring a `subjectAltName` since January 2017.

## The solution

We'll be using [OpenSSL](#) to generate all of our certificates.

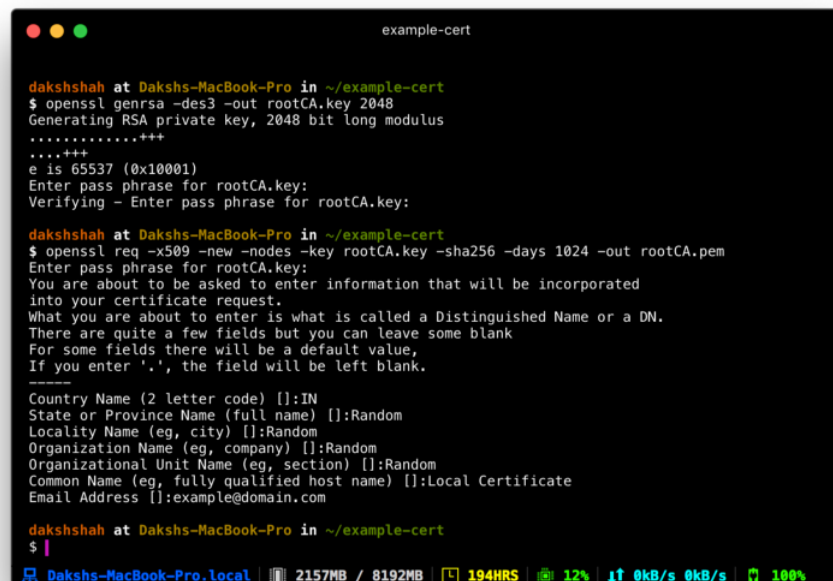
### Step 1: Root SSL certificate

The first step is to create a Root Secure Sockets Layer (SSL) certificate. This root certificate can then be used to sign any number of certificates you might generate for individual domains. If you aren't familiar with the SSL ecosystem, this [article from DNSimple](#) does a good job of introducing Root SSL certificates.

Generate a RSA-2048 key and save it to a file `rootCA.key`. This file will be used as the key to generate the Root SSL certificate. You will be prompted for a pass phrase which you'll need to enter each time you use this particular key to generate a certificate.

You can use the key you generated to create a new Root SSL certificate. Save it to a file named `rootCA.pem`. This certificate will have a validity of 1,024 days. Feel free to change it to any number of days you want. You'll also be prompted for other optional information.

```
openssl req -x509 -new -nodes -key rootCA.key -sha256 -days 1024 -out rootCA.pem
```



```
example-cert

dakshshah at Dakshs-MacBook-Pro in ~/example-cert
$ openssl genrsa -des3 -out rootCA.key 2048
Generating RSA private key, 2048 bit long modulus
.....++
....++
e is 65537 (0x10001)
Enter pass phrase for rootCA.key:
Verifying - Enter pass phrase for rootCA.key:

dakshshah at Dakshs-MacBook-Pro in ~/example-cert
$ openssl req -x509 -new -nodes -key rootCA.key -sha256 -days 1024 -out rootCA.pem
Enter pass phrase for rootCA.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) []:IN
State or Province Name (full name) []:Random
Locality Name (eg, city) []:Random
Organization Name (eg, company) []:Random
Organizational Unit Name (eg, section) []:Random
Common Name (eg, fully qualified host name) []:Local Certificate
Email Address []:example@domain.com

dakshshah at Dakshs-MacBook-Pro in ~/example-cert
$
```

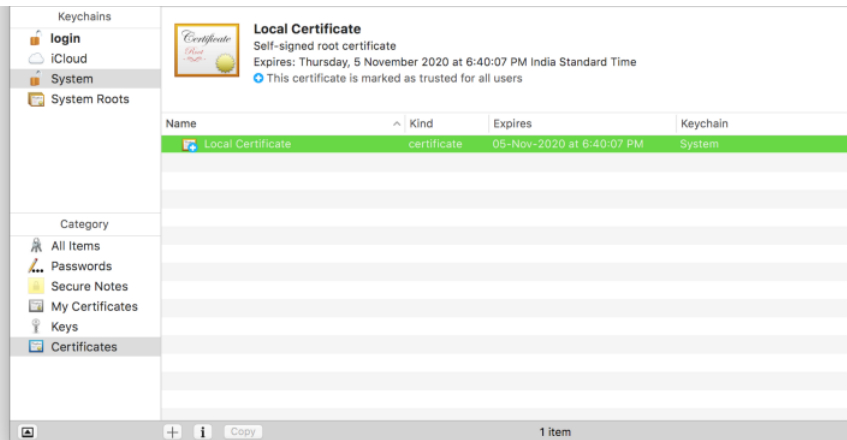
## Step 2: Trust the root SSL certificate

Before you can use the newly created Root SSL certificate to start issuing domain certificates, there's one more step. You need to tell your Mac to trust your root certificate so all individual certificates issued by it are also trusted.

Open Keychain Access on your Mac and go to the Certificates category in your System keychain. Once there, import the `rootCA.pem` using File > Import Items. Double click the imported certificate and change the "When using this certificate:" dropdown to **Always Trust** in the Trust section.

Your certificate should look something like this inside Keychain Access if you've correctly followed the instructions till now.

Learn to code — free 3,000-hour curriculum



## Step 2: Domain SSL certificate

The root SSL certificate can now be used to issue a certificate specifically for your local development environment located at `localhost`.

Create a new OpenSSL configuration file `server.csr.cnf` so you can import these settings when creating a certificate instead of entering them on the command line.

```
[req]
default_bits = 2048
prompt = no
default_md = sha256
distinguished_name = dn

[dn]
C=US
ST=RandomState
L=RandomCity
O=RandomOrganization
OU=RandomOrganizationUnit
emailAddress=hello@example.com
CN = localhost
```

Create a `v3.ext` file in order to create a X509 v3 certificate. Notice how we're specifying `subjectAltName` here.

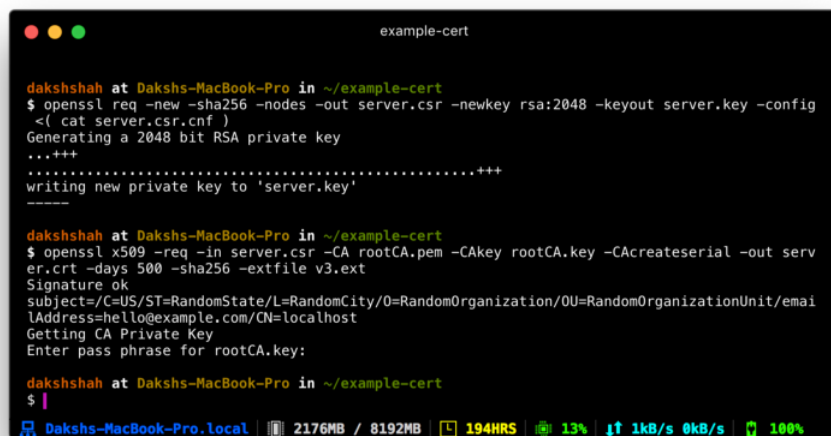
```
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment
subjectAltName = @alt_names

[alt_names]
DNS.1 = localhost
```

Create a certificate key for `localhost` using the configuration settings stored in `server.csr.cnf`. This key is stored in `server.key`.

A certificate signing request is issued via the root SSL certificate we created earlier to create a domain certificate for `localhost`. The output is a certificate file called `server.crt`.

```
openssl x509 -req -in server.csr -CA rootCA.pem -CAkey rootCA.key -CAcreateserial -out server.crt -days 500 -s
```



```
example-cert

dakshshah at Dakshs-MacBook-Pro in ~/example-cert
$ openssl req -new -sha256 -nodes -out server.csr -newkey rsa:2048 -keyout server.key -config
<( cat server.csr.cnf )
Generating a 2048 bit RSA private key
...+++
writing new private key to 'server.key'
-----

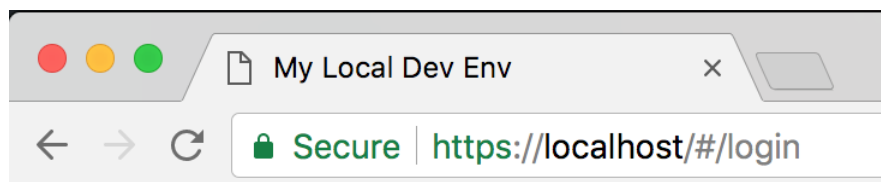
dakshshah at Dakshs-MacBook-Pro in ~/example-cert
$ openssl x509 -req -in server.csr -CA rootCA.pem -CAkey rootCA.key -CAcreateserial -out serv
er.crt -days 500 -sha256 -extfile v3.ext
Signature ok
subject=/C=US/ST=RandomState/L=RandomCity/O=RandomOrganization/OU=RandomOrganizationUnit/emai
lAddress=hello@example.com/CN=localhost
Getting CA Private Key
Enter pass phrase for rootCA.key:

dakshshah at Dakshs-MacBook-Pro in ~/example-cert
$ !
```

## Use your new SSL certificate

You're now ready to secure your `localhost` with HTTPS. Move the `server.key` and `server.crt` files to an accessible location on your server and include them when starting your server.

In an Express app written in Node.js, here's how you would do it. Make sure you do this only for your local environment. **Do not use this in production.**



I hope you found this tutorial useful. If you're not comfortable with running the commands given here by yourself, I've created a set of handy scripts you can run quickly to generate

Learn to code — free 3,000-hour curriculum

*suggestions or feedback. If you'd like to show your appreciation towards any of the work I've done, be it a blog post, an open source project or just a funny tweet, you can [buy me a cup of coffee](#).*



### Daksh Shah

Heads software @tiltbike. Making the web a better place, one line at a time. Dropout (NIT Surat). Self-taught Engineer. Bibliophile.

If you read this far, tweet to the author to show them you care.

[Tweet a thanks](#)

Learn to code for free. freeCodeCamp's open source curriculum has helped more than 40,000 people get jobs as developers.

[Get started](#)

freeCodeCamp is a donor-supported tax-exempt 501(c)(3) nonprofit organization (United States Federal Tax Identification Number: 82-0779546)

Our mission: to help people learn to code for free. We accomplish this by creating thousands of videos, articles, and interactive coding lessons - all freely available to the public. We also have thousands of freeCodeCamp study groups around the world.

Donations to freeCodeCamp go toward our education initiatives, and help pay for servers, services, and staff.

You can [make a tax-deductible donation here](#).

### Trending Guides

What is Docker?	What is STEM?	WordPress for Beginners
TCP/IP Model	JavaScript Void 0	Qualitative VS Quantitative
RTF File	SQL Delete Row	JavaScript Split String
CSS Transition	JavaScript Replace	Accented Letters on Mac
How to Use Instagram?	Python JSON Parser	Windows 10 Product Key
MBR VS GPT	cmd Delete Folder	Google Docs Landscape
FAT32 Format	What is NFC?	Antimalware Executable
Error 503 Code	Content Type JSON	Windows 10 Start Menu
Windows Hosts File	Convert HEIC to JPG	Windows 10 Command Line
Mobi to PDF	Math Random Java	Google Account Recovery

### Our Nonprofit

[About](#) [Alumni Network](#) [Open Source](#) [Shop](#) [Support](#) [Sponsors](#) [Academic Honesty](#)

[Code of Conduct](#) [Privacy Policy](#) [Terms of Service](#) [Copyright Policy](#)