

132

Using Angular CLI to serve over https locally

Richard Russell [Follow](#)

Jun 3, 2018 · 5 min read ★

This is a short description of how to use Angular CLI to serve an Angular Web app over https locally. You might want to do this because you are working with some other services that require secure communication over https; for example, if you are using Facebook login in your app. Once you have followed these steps, you will be able to run and access your local Web app at e.g.

`https://localhost:4200/` for use during development.

Creating the certificate

We will create a self-signed certificate, i.e. one that is signed by ourselves rather than being signed by a trusted Certificate Authority (CA). Consequently, this certificate will not be trusted by default, so we will need to add this to the Trusted Root Certification Authorities store later.

First, let's describe how the certificate is generated. The identity information for the certificate is described in a separate file, `certificate.cnf`. This allows me to specify the 'Subject Alternative Names' parameter for my certificate. This is important because Chrome 58 and later will only check this field for matching the domain name to the certificate, not the common name as was previously used. The `certificate.cnf` file contents are as follows:

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

[Upgrade](#)

```
[dn]
C = GB
ST = London
L = London
O = My Organisation
OU = My Organisational Unit
emailAddress = email@domain.com
CN = localhost
[v3_req]
subjectAltName =
@alt_names
[alt_names]
DNS.1 = localhost
```

132

Next, we can use OpenSSL to generate the certificate. I'm doing this on Windows, so I open Git Bash and run the following command:

```
openssl req -new -x509 -newkey rsa:2048 -sha256 -nodes -keyout
localhost.key -days 3560 -out localhost.crt -config certificate.cnf
```

The above command generates two files: `localhost.key` and `localhost.crt`. These are now ready to use in our Angular CLI config.

Configuring Angular CLI to use https

Now that we have the certificate files, it is simple to configure Angular CLI to use these to serve over https. I place the key and crt files in a separate folder, in this case it's `d:\certificates`. I update my `package.json` file as follows:

```
"start": "ng serve --ssl --ssl-key d:\\certificates\\localhost.key --ssl-cert
d:\\certificates\\localhost.crt"
```

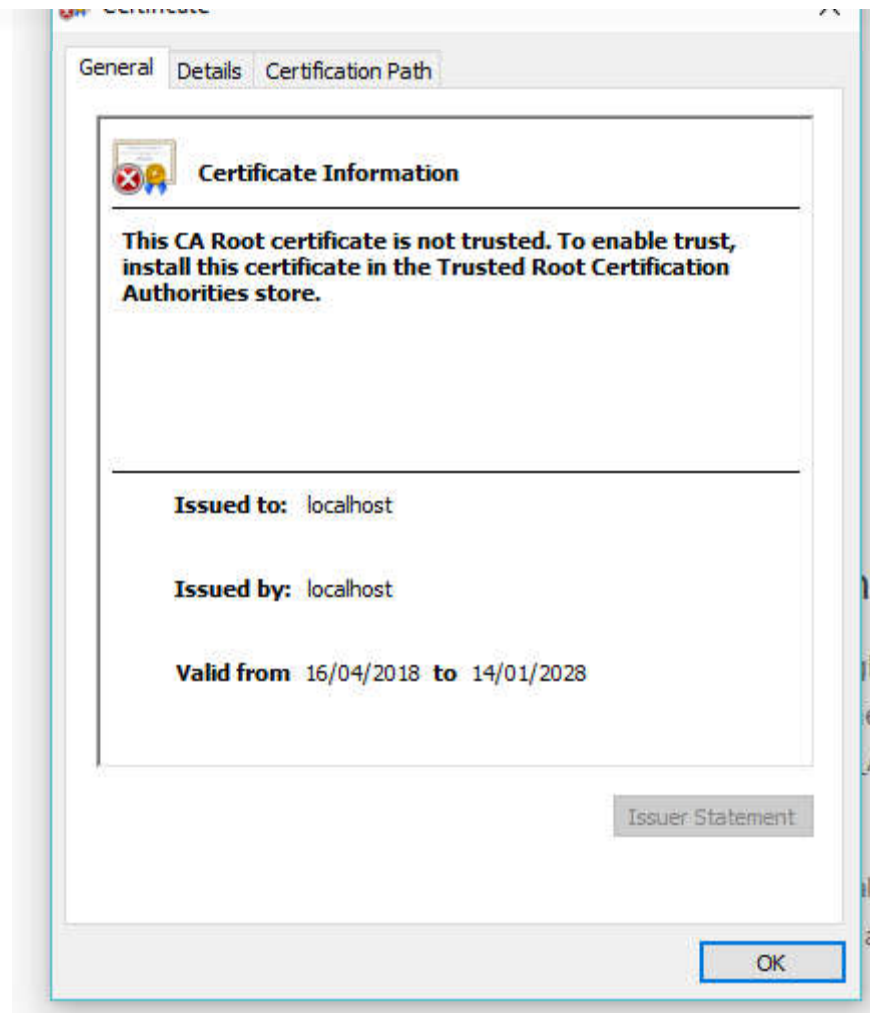
Now running `npm run start` starts serving my Angular app on `https://localhost:4200`, but when I navigate to this URL in Google Chrome I find that I am presented with a warning indicating that the site is not secure. By clicking on the 'Not secure' warning and selecting 'Certificate (Invalid)' I find that Chrome is warning me that the certificate is not trusted:



Upgrade



132

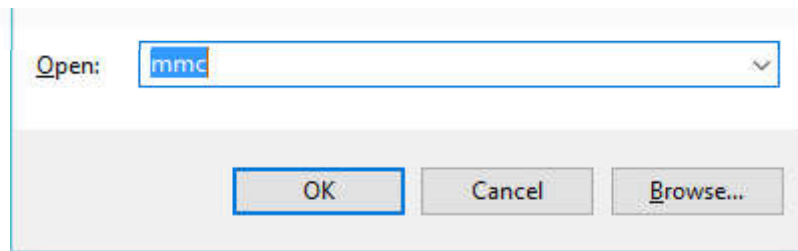


This is because the certificate is self-signed as opposed to being signed by a trusted Certificate Authority. Consequently, I need to add `localhost.crt` to the Trusted Root Certification Authorities store.

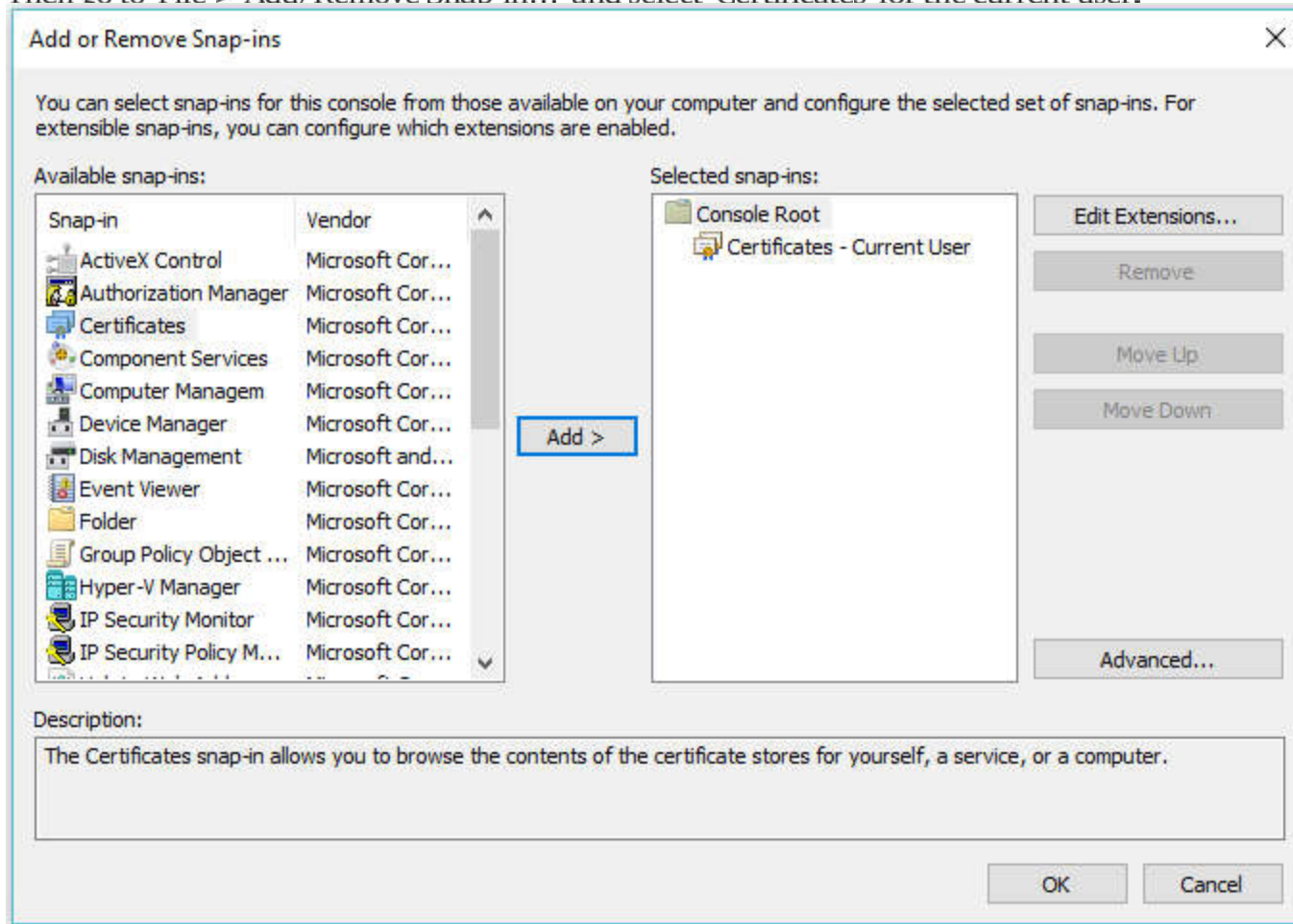
Trusting the certificate on Windows

To add `localhost.crt` to the Trusted Root Certification Authorities store on Windows I need to start 'Microsoft Management Console'. This can be done by pressing <Windows Key> + R or searching for the 'Run' desktop app. Then running 'mmc' like so:

[Upgrade](#)



Then go to 'File > Add/Remove Snap-in...' and select 'Certificates' for the current user:



Once that has been added, you should be able to navigate to:

Console Root
Certificates - Current User








Upgrade




- > Enterprise Trust
- > Intermediate Certification Authorities
- > Active Directory User Object
- > Trusted Publishers
- > Untrusted Certificates

Right-click on 'Certificates' under 'Trusted Root Certification Authorities' and select 'All Tasks > Import...'. Locate your `localhost.crt` file and import it. Once imported, you should be able to find it listed as a trusted certificate:

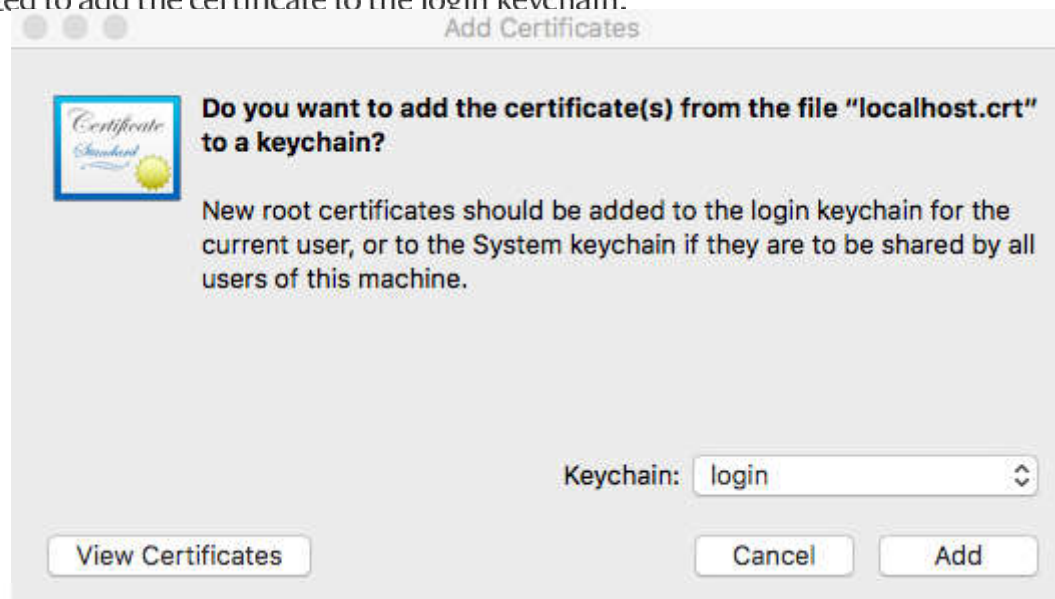
 GTE CyberTrust Global Root	GTE CyberTrust Global Root	14/08/2018	Secure Email, Client...	DigiCert Global Root
 Hotspot 2.0 Trust Root CA - 03	Hotspot 2.0 Trust Root CA - 03	08/12/2043	Server Authenticati...	Hotspot 2.0 Trust R...
 localhost	localhost	14/01/2028	<All>	<None>
 Microsoft Authenticode(tm) Ro...	Microsoft Authenticode(tm) Root...	01/01/2000	Secure Email, Code ...	Microsoft Authenti...
 Microsoft Root Authority	Microsoft Root Authority	31/12/2020	<All>	Microsoft Root Aut...

Close Microsoft Management Console (you do not need to save the console). Then restart Chrome. You should now find that the certificate is trusted when you load `https://localhost:4200/`:

 Secure | <https://localhost:4200>

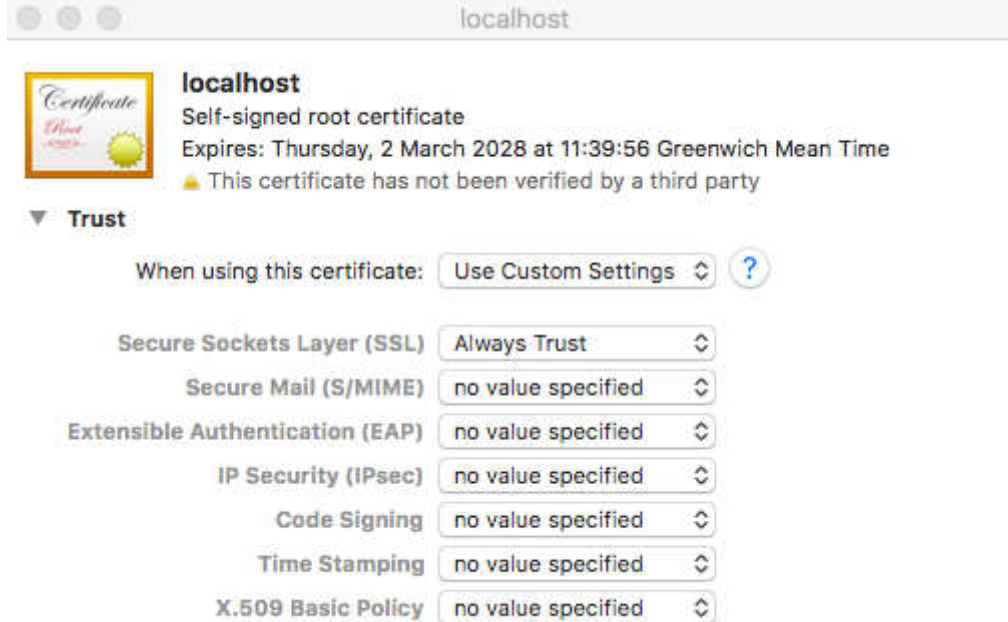
Trusting the certificate on Mac OS X

It is slightly simpler to trust the certificate on OS X. Double-click the `localhost.crt` file and you will be prompted to add the certificate to the login keychain:



...	iTunes iAd password	iTunes iAd password	31 Aug 2015, 15:52:58	--	login
+	localhost	certificate	--	2 Mar 2028, 11:39:56	login
...	MetadataKeychain	application password	9 Oct 2016, 00:38:37	--	login
...	ProtectedCloudStorage	application password	14 Jun 2015, 19:42:08	--	login

Select 'localhost' and right-click to select 'Get Info' from the context menu. Then expand the 'Trust' triangle. You should then be able to select to 'Always Trust' the certificate for SSL:



You will most likely be prompted to enter your admin password to make this change. After the change has been applied, you should see the warning disappear in Chrome when reloading the page:



That's all! Hopefully, that was easy to setup and now you can work locally with services that require the app to be served over https. It's important to stress that this is only for local development! You should use certificates that are signed by a trusted Certificate Authority in public facing apps.

References

1. Certificate Authority on Wikipedia: https://en.wikipedia.org/wiki/Certificate_authority
2. Angular CLI wiki for ng serve: <https://github.com/angular/angular-cli/wiki/serve>
3. Open SSL documentation: <https://www.openssl.org/docs/man1.0.2/apps/openssl-req.html>



Upgrade



5. Trusting certificates in OS X Sierra: https://support.apple.com/kb/ph25443?locale=en_GB

Ssl Angular Angular Cli



132 claps



132



WRITTEN BY

Richard Russell

Follow

[See responses \(2\)](#)

More From Medium

Related reads

Related reads

Related reads



Upgrade



Setup a Proxy for API Calls for Your Angular CLI App



Spencer Feng in...
Apr 14, 2018 · 2 min...



111



Angular HTTP Interceptors: What are they and how to use them



Denis Nuțiu in...
May 23, 2018 · 2 mi...



467



Angular 6 Url Parameters



Christopher Jeffery i...
Aug 25, 2018 · 3 min...



1K



132