How do I allow HTTPS for Apache on localhost?

Asked 10 years, 3 months ago Active 23 days ago Viewed 272k times

204

I was asked to set up HTTPS with a self-signed cert on Apache on localhost, but how do I actually do that? I have no idea at all.

apache ssl https localhost

111

Share Improve this question Follow

edited Aug 1 '14 at 10:05
TRiG
9,137 6 49 97

asked Nov 19 '10 at 3:37
KennC.
2,935 6 18 18

1 Use Serveo! `ssh -R youruniquesubdomain:80:localhost:3000 serveo.net` Slap in your subdomain and port number and you ready to go on `https://youruniquesubdomain.serveo.net` – totymedli Jul 11 '18 at 13:15

@totymedli I get ssh: connect to host serveo.net port 22: Connection refused – Timo Oct 21 '20 at 10:49

2 @Timo Seems like Serveo is dead, but localhost.run does the same: `ssh -R 80:localhost:8080 ssh.localhost.run` – totymedli Oct 22 '20 at 11:59

@totymedli, awesome answer-comment! I had this going in a couple of minutes, didn't even read any of those verbose answers, lol. – Octopus Jan 21 at 4:32

16 Answers

Active | Oldest | Votes

137

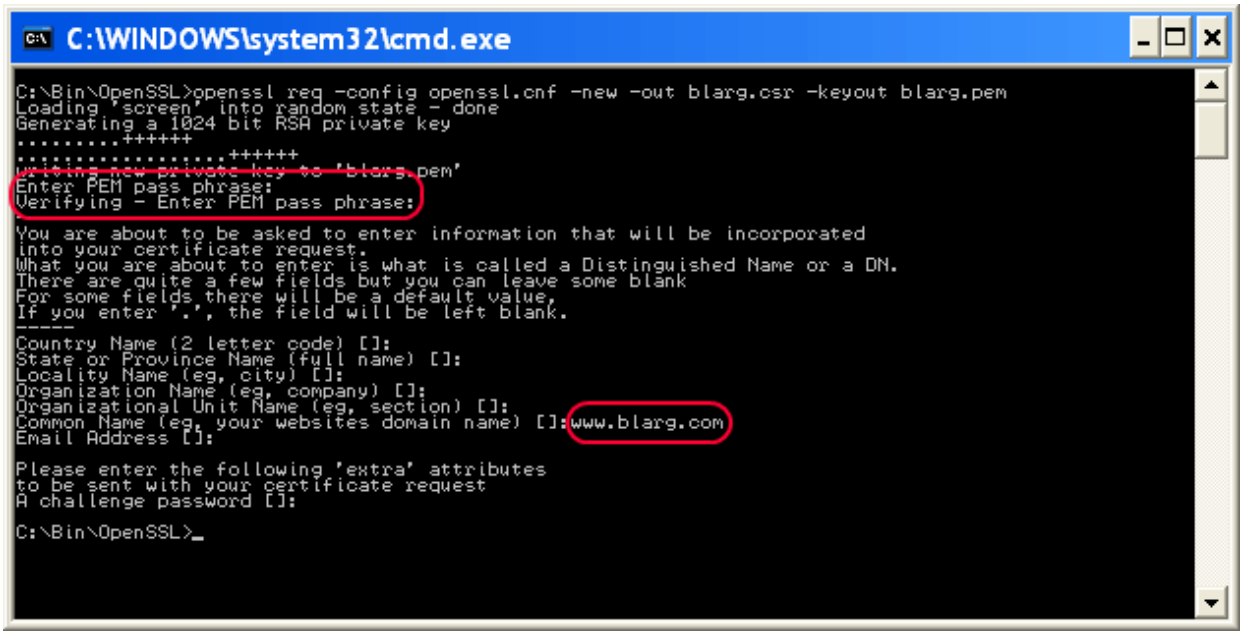
I've just attempted this - I needed to test some development code on my **localhost Apache on Windows**. This was WAAAY more difficult than it should be. But here are the steps that managed to work after much hairpulling...

I found that my Apache install comes with **`openssl.exe`** which is helpful. If you don't have a copy, you'll need to download it. My copy was in `Apache2\bin` folder which is how I reference it below.

Steps:

1. Ensure you have write permissions to your Apache conf folder
2. Open a command prompt in `Apache2\conf` folder
3. Type
   ```
   ..\bin\openssl req -config openssl.cnf -new -out blarg.csr -keyout blarg.pem
   ```
4. You can leave all questions blank except:

- Common Name: the hostname of your server

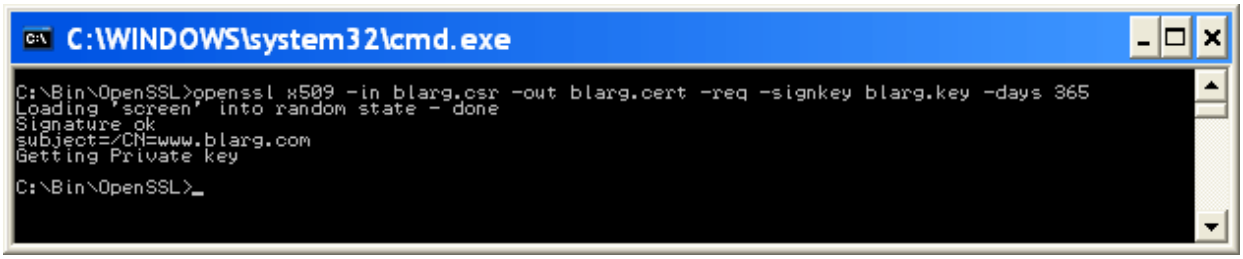


5. When that completes, type

   ```
   ..\bin\openssl rsa -in blarg.pem -out blarg.key
   ```

6. Generate your self-signed certificate by typing:

   ```
   ..\bin\openssl x509 -in blarg.csr -out blarg.cert -req -signkey blarg.key -days 365
   ```



7. Open Apache's `conf\httpd.conf` file and ensure SSL module is enabled - there should be no hash at the start of this line:

   ```
   LoadModule ssl_module modules/mod_ssl.so
   ```

   ```
   #LoadModule rewrite_module modules/mod_rewrite.so
   LoadModule setenvif_module modules/mod_setenvif.so
   #LoadModule speling_module modules/mod_speling.so
   #LoadModule status_module modules/mod_status.so
   #LoadModule unique_id_module modules/mod_unique_id.so
   LoadModule userdir_module modules/mod_userdir.so
   #LoadModule usertrack_module modules/mod_usertrack.so
   #LoadModule vhost_alias_module modules/mod_vhost_alias.so
   LoadModule ssl_module modules/mod_ssl.so
   ```

8. Some Apache installations place the SSL config in a separate file. If so, ensure that the SSL conf file is being included. In my case I had to uncomment this line:

   ```
   Include conf/extra/httpd-ssl.conf
   ```

9. In the SSL config `httpd-ssl.conf` I had to update the following lines:

   - Update

```
SSLSessionCache "shmcb:C:/Progra\~2/Zend/Apache2/logs/ssl_scache(512000)"
```

(The brackets in the path confuse the module, so we need to escape them)

- `DocumentRoot` - set this to the folder for your web files
- `ServerName` - the server's hostname
- `SSLCertificateFile "conf/blarg.cert"`
- `SSLCertificateKeyFile "conf/blarg.key"`

```
77  ##
78  ## SSL Virtual Host Context
79  ##
80
81  <VirtualHost _default_:443>
82
83  #   General setup for the virtual host
84  DocumentRoot "c:/Inetpub/wwwroot"
85  ServerName www.blarg.com:443
86  ServerAdmin webmaster@blarg.com
87  ErrorLog logs/error_log
88  TransferLog logs/access_log
99  #   Server Certificate:
100 #   Point SSLCertificateFile at a PEM encoded certificate.  If
101 #   the certificate is encrypted, then you will be prompted for a
102 #   pass phrase.  Note that a kill -HUP will prompt again.  Keep
103 #   in mind that if you have both an RSA and a DSA certificate you
104 #   can configure both in parallel (to also allow the use of DSA
105 #   ciphers, etc.)
106 SSLCertificateFile conf/ssl/blarg.cert
107 #SSLCertificateFile conf/ssl.crt/server-dsa.crt
108
109 #   Server Private Key:
110 #   If the key is not combined with the certificate, use this
111 #   directive to point at the key file.  Keep in mind that if
112 #   you've both a RSA and a DSA private key you can configure
113 #   both in parallel (to also allow the use of DSA ciphers, etc.)
114 SSLCertificateKeyFile conf/ssl/blarg.key
115 #SSLCertificateKeyFile conf/ssl.key/server-dsa.key
```

10. Restart Apache.
11. Try loading `https://localhost/` in your browser.

Hopefully you made it this far. Feel free to update this post with any other helpful info.

(Screenshots courtesy of Neil Obremski and his helpful article - although now quite out-of-date.)

Share Improve this answer Follow edited Apr 2 '14 at 0:07 community wiki 3 revs Simon

2 Thanks. I also had to edit the ErrorLog, TransferLog and CustomLog directives to valid paths otherwise Apache wouldn't start. – Tamlyn Nov 29 '12 at 17:32

1 For some reason, instructions do not work any longer or incomplete – Jacobian Dec 3 '14 at 13:52

6 I had to uncomment the following as well in my httpd.conf for it to work: LoadModule socache_shmcb_module modules/mod_socache_shmcb.so – erik Sep 21 '16 at 11:33

5 In order to generate the .pem and .key files, I had to set 2 environment variables at step 2 : set OPENSSL_CONF=C:\path\to\apache\Apache2.4.4\conf\openssl.cnf set RANDFILE=C:\path\to\apache\Apache2.4.4\conf\.rnd – eosphere Apr 9 '17 at 11:27

58

I use ngrok (https://ngrok.com/) for this. ngrok is a command line tool and create a tunnel for localhost. It creates both http and https connection. After downloading it, following command needs to be run :

```
ngrok http 80
```

(In version 2, the syntax is : ngrok http 80 . In version 2, any port can be tunneled.)

After few seconds, it will give two urls :

```
http://a_hexadecimal_number.ngrok.com
https://a_hexadecimal_number.ngrok.com
```

Now, both the urls point to the localhost.

Share Improve this answer Follow

edited Oct 11 '16 at 8:07

j0k
21.8k 28 74 84

answered Apr 23 '14 at 12:14

sudip
2,433 25 39

1 @sudip, Does the opensource code of ngrok works in such a way that we can host this on our own server without modification? If not, it's pretty much a showstopper because it's **not ok** to redirect users requests to an external host like ngrok. – Pacerier Mar 15 '15 at 18:38

2 @Pacerier I dont intend to use it on server. I use it on localhost (Bcz my network provider gives me a dynamic IP). I used it first time for paypal IPN testing and it worked perfectly. I wonder why someone will use it on server and for what purpose. – sudip Mar 16 '15 at 21:53

@sudip, The purpose is obvious, To allow code that works on HTTP to also work with HTTPS with no extra coding needed. – Pacerier Mar 19 '15 at 15:10

1 Though this is useful, it seems incredibly insecure to allow access to your dev machine to the open internet. Using something like this would get you fired at a security conscious employer. – Andy M Apr 5 '16 at 14:47

@YumYumYum . It was completely free before in V 1. But, http and https ports are still free in v 2 (dont know whether any port restriction is there is free plan). Check the free plan here : ngrok.com/product#pricing – sudip Apr 15 '16 at 17:32

25

here is simplest way to do this

first copy these server.crt & server.key files (find in attachment) into your apache/conf/ssl directory

```
Listen 80
Listen 443

NameVirtualHost *:80
NameVirtualHost *:443

<VirtualHost *:443>
    DocumentRoot "d:/wamp/www"  #your wamp www root dir
    ServerName localhost
    SSLEngine on
    SSLCertificateFile "d:/wamp/bin/apache/Apache2.4.4/conf/ssl/server.crt"
    SSLCertificateKeyFile "d:/wamp/bin/apache/Apache2.4.4/conf/ssl/server.key"
</VirtualHost>
```

Share Improve this answer Follow edited Dec 7 '17 at 9:30 answered Nov 17 '14 at 7:04

Anil Gupta
2,229 3 21 30

3 I had to enable also the module LoadModule ssl_module libexec/apache2/mod_ssl.so in the (/etc/apache2/httpd.conf) – Alexey May 8 '15 at 9:25

17 I wonder how safe/dangerous is downloading *.crt *.key files from untrusted source instead of generating your own. – Petr Peller Apr 12 '16 at 16:22

4 @PetrPeller we are setting up https for local development so why wonder for safe/dangerous – Anil Gupta Apr 13 '16 at 7:59

7 An explanation how to generate those files would be great. Because downloading files from an unknown source is a bad practise, but also because that kind of links will break at some point. – Stephan Vierkant Apr 10 '17 at 19:28

2 This tutorial is fine digitalocean.com/community/tutorials/… – Dhiraj Feb 9 '18 at 12:45

18

In order to protect the security of information being sent to and from your web server, it's a good idea to enable encryption of the communication between clients and the server. This is often called **SSL**.

So let's set up HTTPS with a self-signed certificate on Apache2. I am going to list the steps which you should follow:

- Install apache2 web-server on your machine. For linux machine open the terminal and type

  sudo apt-get install apache2

- After successful installation check the status of apache2 service by executing command

  sudo service apache2 status

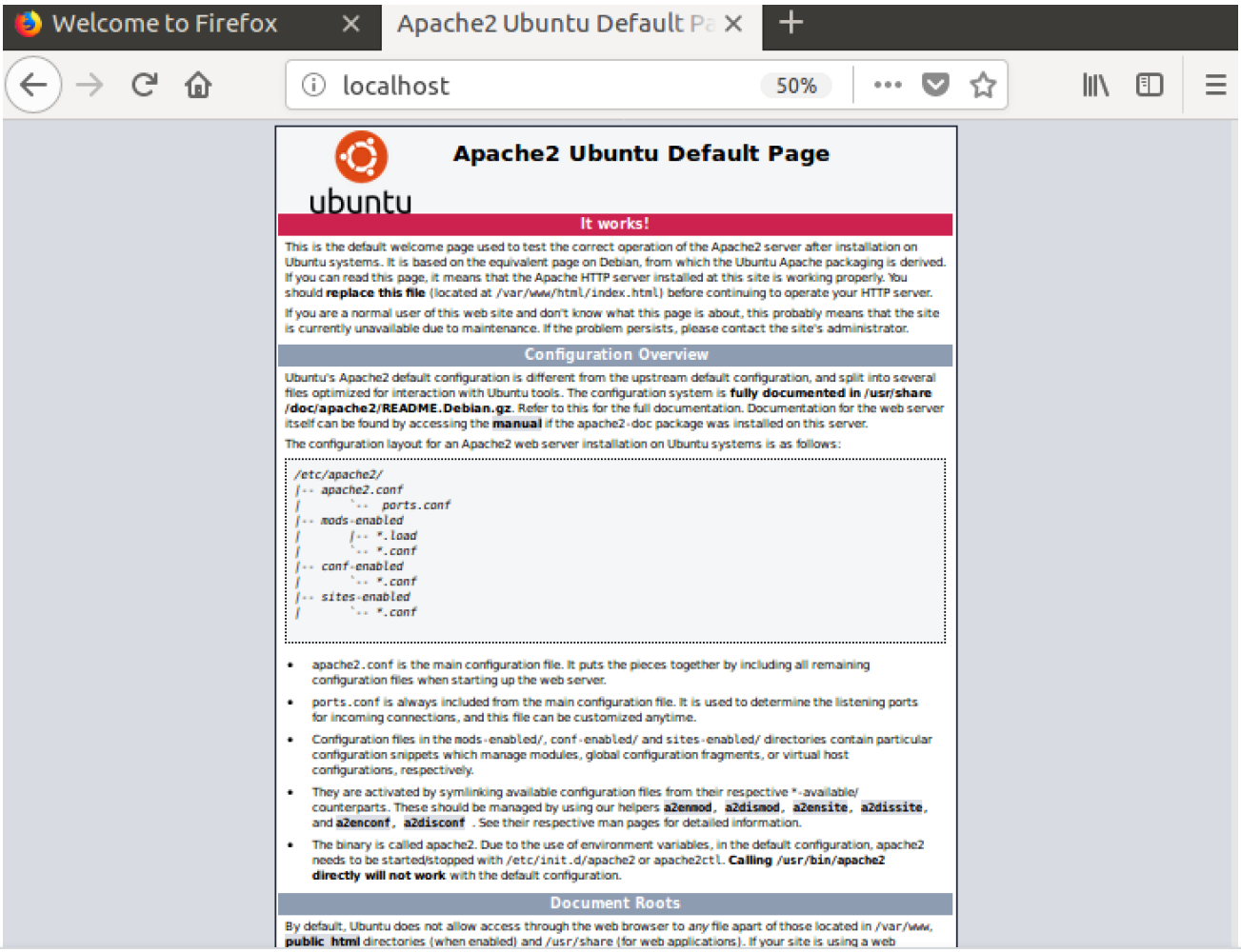
```
dinesh@dinesh-VirtualBox:~$ sudo service apache2 status
● apache2.service - LSB: Apache2 web server
   Loaded: loaded (/etc/init.d/apache2; bad; vendor preset: enabled)
  Drop-In: /lib/systemd/system/apache2.service.d
           └─apache2-systemd.conf
   Active: active (running) since Sat 2018-03-24 17:13:42 IST; 1min 28s ago
     Docs: man:systemd-sysv-generator(8)
  CGroup: /system.slice/apache2.service
           ├─19300 /usr/sbin/apache2 -k start
           ├─19303 /usr/sbin/apache2 -k start
           └─19304 /usr/sbin/apache2 -k start

Mar 24 17:13:40 dinesh-VirtualBox systemd[1]: Starting LSB: Apache2 web server..
Mar 24 17:13:40 dinesh-VirtualBox apache2[19278]:  * Starting Apache httpd web s
Mar 24 17:13:41 dinesh-VirtualBox apache2[19278]: AH00558: apache2: Could not re
Mar 24 17:13:42 dinesh-VirtualBox apache2[19278]:  *
Mar 24 17:13:42 dinesh-VirtualBox systemd[1]: Started LSB: Apache2 web server.
lines 1-16/16 (END)
```

- Navigate to browser and type

  http://localhost:80

Verify that you get default page for apache2 like this.



- For encrypting a web connection we need certificate from CA (certificate authority) or we can use self signed certificates. Let's create a self signed certificate using the following command.

  openssl req -x509 -newkey rsa:2048 -keyout mykey.key -out mycert.pem -days 365 -nodes

Please fill the information accordingly as shown below.

```
dinesh@dinesh-VirtualBox:~$ openssl req -x509 -newkey rsa:4096 -keyout mykey.pem
 -out mycert.pem -days 365 -nodes
Generating a 4096 bit RSA private key
.....................................++
.....++
writing new private key to 'mykey.pem'
-----
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) [AU]:IN
State or Province Name (full name) [Some-State]:Maharastra
Locality Name (eg, city) []:Navi Mumbai
Organization Name (eg, company) [Internet Widgits Pty Ltd]:abc.com
Organizational Unit Name (eg, section) []:abc
Common Name (e.g. server FQDN or YOUR name) []:localhost
Email Address []:abc@gmail.com
dinesh@dinesh-VirtualBox:~$
```

mykey.key and **mycert.pem** should be created in your present working directory.

- It would be nice we if move certificates and keys at a common place and it will be easy for apache2 web server to find them. So let's execute the following commands

  sudo cp mycert.pem /etc/ssl/certs

  sudo cp mykey.key /etc/ssl/private

- Let's enable the SSL mode on your server

  sudo a2enmod ssl

It should output like this

```
dinesh@dinesh-VirtualBox:~$ sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create s
elf-signed certificates.
To activate the new configuration, you need to run:
  service apache2 restart
```

- Let's configure apache2 to use self signed certificate and key which we have generated above.

  sudo vi /etc/apache2/sites-available/default-ssl.conf

Please find these two lines and replace them with your cert and key paths.

Initial

```
#   SSLCertificateFile directive is needed.
SSLCertificateFile      /etc/ssl/certs/ssl-cert-snakeoil.pem
SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
```

Final

```
SSLCertificateFile      /etc/ssl/certs/mycert.pem
SSLCertificateKeyFile /etc/ssl/private/mykey.key
```

- Enable the site

  cd /etc/apache2/sites-available/

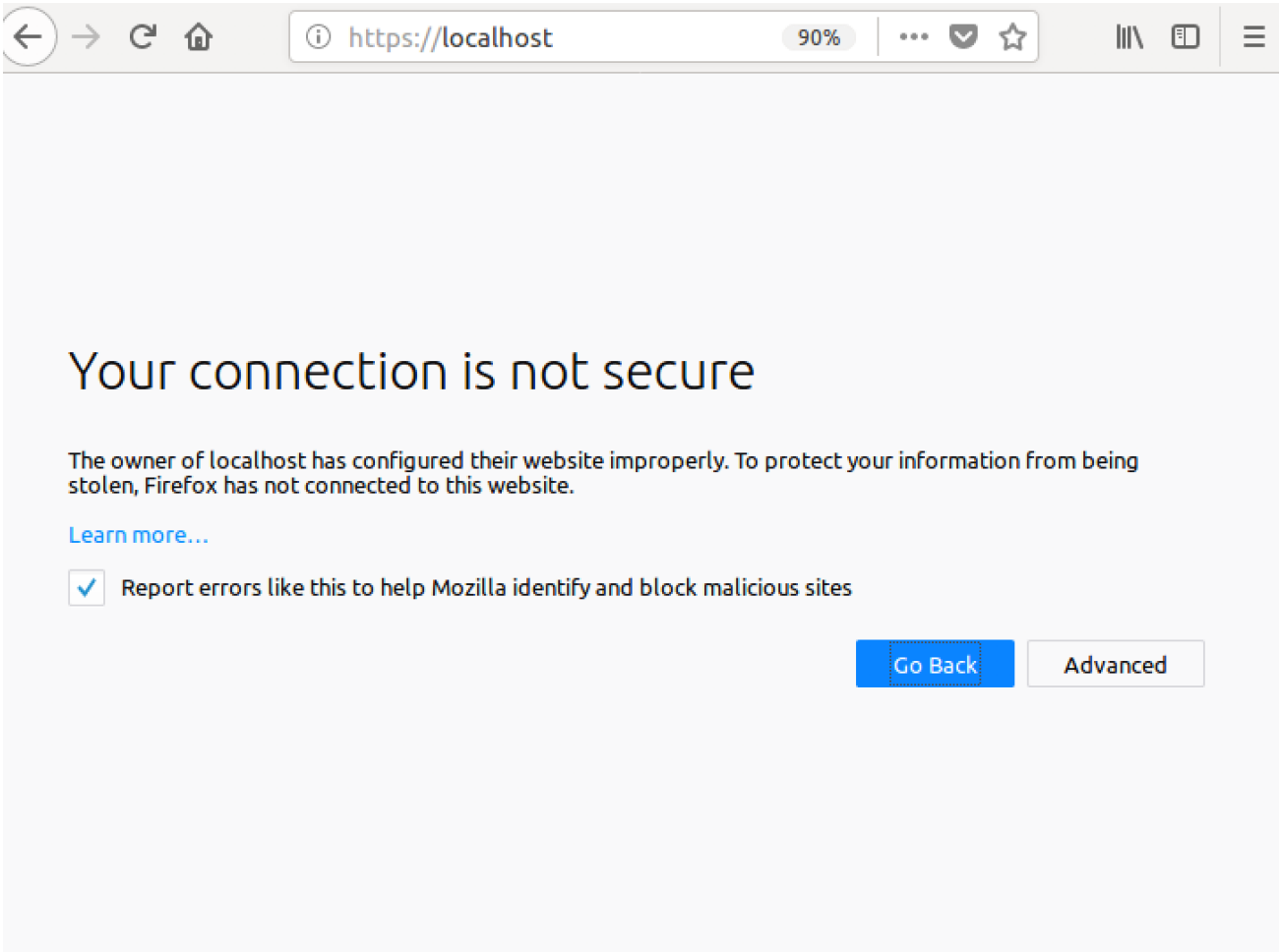
  sudo a2ensite default-ssl.conf

- Restart the apache2 service

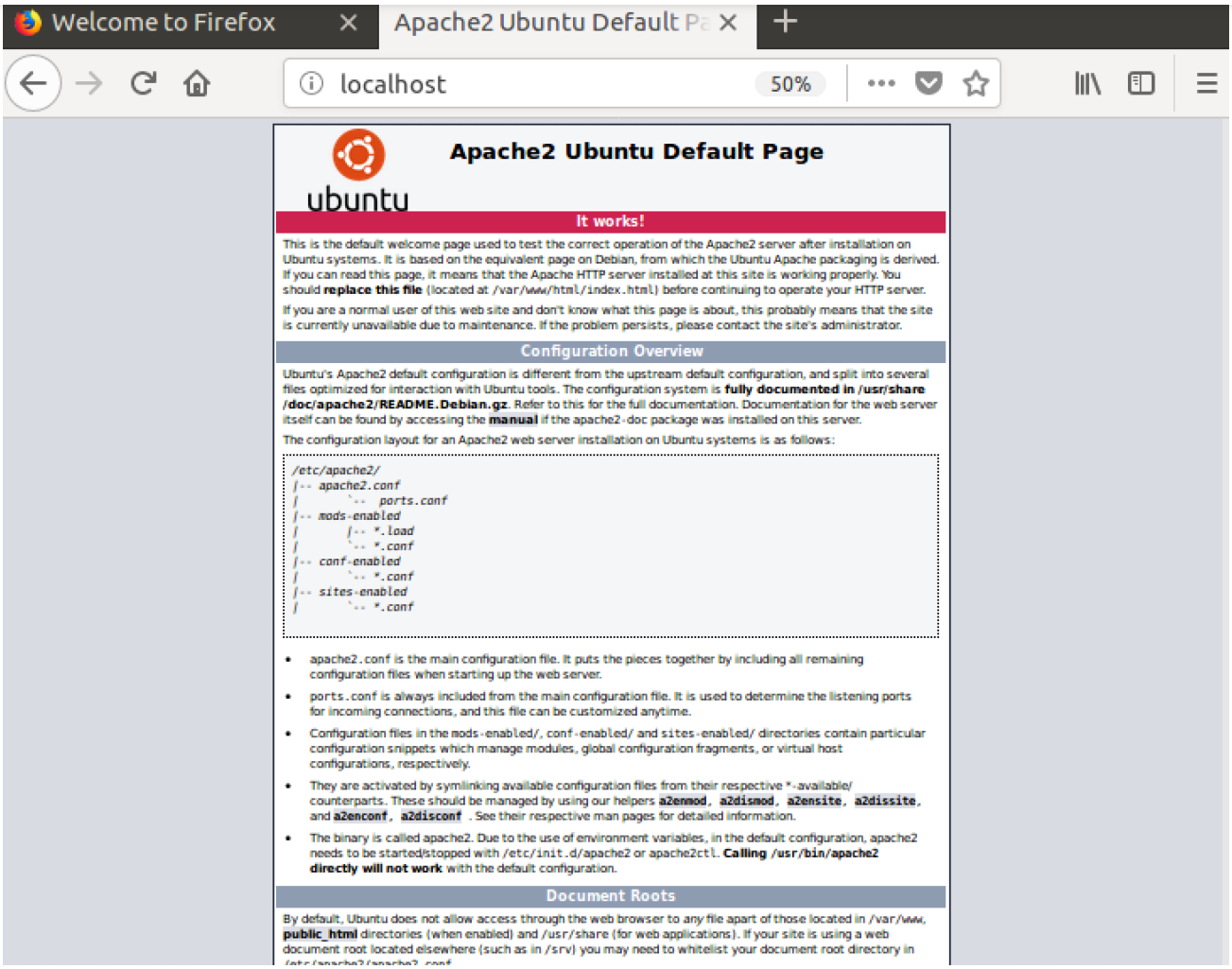
  sudo service apache2 restart

- Verify the apache2 web-server on HTTPS. Open your browser again and type

It should output something like this with a warning that page you are about to view is not secure because we have configured the server with self-signed certificate.



- Congratulations you have configured your apache2 with HTTPS endpoint , now click on **advanced** --> **add exception** --> **confirm security exception** , you will see the default page again.



Share Improve this answer Follow

edited Apr 10 '18 at 11:38

Elshan

5,832 3 53 90

answered Mar 24 '18 at 13:12

Dinesh Kumar

476 3 14

I prefer not to edit any config file if I can, so I left `default-ssl.conf` as it is. I was about to rename `mycert` to `ssl-cert-snakeoil` but this file already exists so I just used that! So I was able to safely skip two steps on Debian. – Rolf Jun 7 '18 at 9:09

@Rolf I agree with you but in production, it is always the case use a new certificate and key. So just to show how they can be created, I have added 2 additional steps for Debian. Thanks :) – Dinesh Kumar Jan 28 '20 at 12:08

Is there any way I can add my self-signed certificate as a certificate authority, to avoid seeing warnings? – Aaron Franke Jul 13 '20 at 1:13

10

Windows + Apache 2.4, for example:

1. uncomment ssl_module in your `httpd.conf` file.

```
LoadModule ssl_module modules/mod_ssl.so
```

```
Listen 80
Listen 443
```

3. uncomment Include Virtual hosts in your `httpd.conf` file.

```
# Virtual hosts
Include conf/extra/httpd-vhosts.conf
```

4. add VirtualHost in your `conf/extra/httpd-vhosts.conf`

```
<VirtualHost _default_:443>
    DocumentRoot "D:/www"  #your site directory path
    ServerName localhost
    #ServerAlias localhost.com localhost2.com
    SSLEngine on
    SSLCertificateFile "${SRVROOT}/conf/ssl/server.crt"
    SSLCertificateKeyFile "${SRVROOT}/conf/ssl/server.key"
    <Directory "D:/www">
        Options -Indexes +FollowSymLinks +ExecCGI
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
```

only the port number `443` and `SSL.....` lines are different from normal http config.

save you config file and restart apache service. then you can visit https://localhost/

The web browser will warn you that it's unsafe at the first time, just choose go on.

Share Improve this answer Follow

answered Sep 2 '15 at 9:27

cuixiping
18.1k 4 68 86

This has worked for me on XP Home, Apache 2.4. Copied the 2 certificate files from the previous post (by Anil Gupta). Uncommented mod_ssl and included httpd-vhosts.conf in httpd.conf, added Anil Gupta's VirtualHost directive (with some paths adjusted) in httpd-vhosts.conf. – jogi99 Apr 19 '16 at 9:36

8

It's actually quite easy, assuming you have an openssl installation handy. (What platform are you on?)

Assuming you're on linux/solaris/mac os/x, Van's Apache SSL/TLS mini-HOWTO has an excellent walkthrough that I won't reproduce here.

However, the executive summary is that you have to create a self-signed certificate. Since you're running apache for localhost presumably for development (i.e. not a public web server), you'll know that you can trust the self-signed certificate and can ignore the warnings that your browser

edited Sep 24 '13 at 11:24

reevesy

3,334 1 24 22

answered Nov 19 '10 at 3:43

Pete Clark

576 4 8

Hi, I'm working on Windows OS. And as for the self signed certificate, do I have to download it or by any other means? – KennC. Nov 19 '10 at 3:47

3 Nope. You'll make the self-signed cert yourself. Do you have the apache httpd + ssl setup? You'll need the ssl in order to do this. This site: rubayathasan.com/tutorial/apache-ssl-on-windows has good info on getting ssl going on windows. You'll be doing some command-line work, but that's good for you anyway. :-) – Pete Clark Nov 19 '10 at 3:51

3 The links is dead :(– kpuccino Jun 1 '17 at 17:55

Yeah - it does look to be dead. That's the internet for you... However, the link to the CentOS Wiki below referenced by @kayakinkoder is also good: wiki.centos.org/HowTos/Https If you're on a mac, this writeup also looks reasonable: gist.github.com/nrollr/4daba07c67adcb30693e – Pete Clark Jun 3 '17 at 2:58

@PeteClark do you have for windows – Adam Mudianto Sep 5 '20 at 12:35

4

This should be work Ubuntu, Mint similar with Apache2

It is a nice guide, so following this

https://www.digitalocean.com/community/tutorials/how-to-create-a-ssl-certificate-on-apache-for-ubuntu-14-04

and leaving your ssl.conf like this or similar similar

```
<VirtualHost _default_:443>
        ServerAdmin your@email.com
        ServerName localhost
        ServerAlias www.localhost.com

        DocumentRoot /var/www


    SSLEngine on
    SSLCertificateFile /etc/apache2/ssl/apache.crt
    SSLCertificateKeyFile /etc/apache2/ssl/apache.key
```

you can get it.

Hope this help for linuxer

Share Improve this answer Follow

answered Dec 11 '14 at 15:36

ackuser

4,298 5 32 43

4

I'm posting this answer since I struggled with this myself and Chrome updated their security with requiring ***Subject Alternative Name*** which a lot of posts do not have as it was not required when they were posted as an answer. I'm assuming that WAMP is already installed.

STEP 1

Download OpenSSL Light and install

STEP 2 (Optional)
Although this part is optional, but it makes it easier later to execute commands. If you skip this step, you'll have to provide full path to openssl.exe where you will execute the command. If you prefer to set it then update the openssl.exe path in Environment Variables.

> Environment Variables -> System Variables -> Path -> Edit -> New -> c:\Program Files\OpenSSL-Win64\bin

STEP 3
Create a folder named **"key"** in the `c:/wamp64/bin/apache/apache2.4.27(your version number)/conf/` directory.

Create configuration file for your CA ***MyCompanyCA.cnf*** with contents (you can change it to your needs):

```
[ req ]
distinguished_name  = req_distinguished_name
x509_extensions     = root_ca

[ req_distinguished_name ]
countryName             = Country Name (2 letter code)
countryName_min         = 2
countryName_max         = 2
stateOrProvinceName     = State or Province Name (full name)
localityName            = Locality Name (eg, city)
0.organizationName      = Organization Name (eg, company)
organizationalUnitName  = Organizational Unit Name (eg, section)
commonName              = Common Name (eg, fully qualified host name)
commonName_max          = 64
emailAddress            = Email Address
emailAddress_max        = 64

[ root_ca ]
basicConstraints            = critical, CA:true
```

Create the extensions configuration file ***MyCompanyLocalhost.ext*** for your web server certificate:

```
[alt_names]
DNS.1   = localhost
DNS.2   = mycy.mycompany.com
```

STEP 4
Execute these commands in the given order to generate the key and certificates:

```
openssl req -x509 -newkey rsa:2048 -out MyCompanyCA.cer -outform PEM -keyout MyCompanyCA.pvk -
days 10000 -verbose -config MyCompanyCA.cnf -nodes -sha256 -subj "/CN=MyCompany CA"
openssl req -newkey rsa:2048 -keyout MyCompanyLocalhost.pvk -out MyCompanyLocalhost.req -subj
/CN=localhost -sha256 -nodes
openssl x509 -req -CA MyCompanyCA.cer -CAkey MyCompanyCA.pvk -in MyCompanyLocalhost.req -out
MyCompanyLocalhost.cer -days 10000 -extfile MyCompanyLocalhost.ext -sha256 -set_serial 0x1111
```

As a result, you will have **MyCompanyCA.cer**, **MyCompanyLocalhost.cer** and **MyCompanyLocalhost.pvk** files.

STEP 5
Install **MyCompanyCA.cer** under

> Control Panel -> Manage User Certificates -> Trusted Root Certification Authorities -> Certificates

To install **MyCompanyLocalhost.cer** just double click it.

STEP 6
Open `c:/wamp64/bin/apache/apache2.4.27(your version number)/conf/httpd.conf` and **un-comment (remove the #)** the following 3 lines:

```
LoadModule ssl_module modules/mod_ssl.so
Include conf/extra/httpd-ssl.conf
LoadModule socache_shmcb_module modules/mod_socache_shmcb.so
```

STEP 7
Open `c:/wamp64/bin/apache/apache2.4.37/conf/extra/httpd-ssl.conf` and change all the parameters to the ones shown below:

```
Directory "c:/wamp64/www"
DocumentRoot "c:/wamp64/www"
ServerName localhost:443
ServerAdmin admin@example.com
```

```
SSLCertificateKeyFile "c:/wamp64/bin/apache/apache2.4.27/conf/key/MyCompanyLocalhost.pvk"
SSLSessionCache "shmcb:c:/wamp64/bin/apache/apache2.4.27/logs/ssl_scache(512000)"
CustomLog "c:/wamp64/bin/apache/apache2.4.27/logs/ssl_request.log" \
          "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
```

Note: *This is the tricky part. If you make any small mistake while editing this file, SSL won't work. Make a copy of it before you edit it.*

STEP 8
Restart Wamp and Chrome. Localhost is now secure: https://localhost

Share Improve this answer Follow edited Jan 28 at 23:50 answered Jun 28 '19 at 4:59

CodeWarrior
4,108 5 21 40

Do you have a guide for Linux? – Aaron Franke Jul 13 '20 at 0:51

Unfortunately, I don't. – CodeWarrior Jul 13 '20 at 1:13

Could you possibly present a sample script? It'd be great to be able to do an automated install of this using Powershell or BAT. Thanks. – Ifedi Okonkwo Dec 6 '20 at 10:25

```
 unable to find 'distinguished_name' in config problems making Certificate Request
18140:error:0E06D06C:configuration file routines:NCONF_get_string:no
value:crypto\conf\conf_lib.c:273:group=req name=distinguished_name
```
– Scott Fleming Feb 16 at 15:31

2

It's very simple,

just run the following commands

```
sudo a2enmod ssl

sudo service apache2 restart

sudo a2ensite default-ssl.conf
```

That's it, you are done.

If you want to force SSL (to use https always), edit the file:

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

and add this one line

```
        Redirect "/" "https://your_domain_or_IP/"

        . . .
</VirtualHost>
```

then restart again

```
sudo service apache2 restart
```

Share Improve this answer Follow

answered Jan 24 '17 at 7:06

Moh .S

1,396 14 17

And you need to run `systemctl reload apache2` after `sudo a2ensite default-ssl.conf` . – untill Mar 14 '19 at 10:23

2

Running Apache on Windows 10 here. I couldn't get Chrome to trust the certificate made in the top answer by Simon. What I ended up doing was using PowerShell to generate a self signed certificate.

Step 1 - Generate Self-Signed certificate

In PowerShell

`New-SelfSignedCertificate -DnsName "localhost" -CertStoreLocation "cert:\LocalMachine\My"` [1]

Step 2 - Configure and Export Certificate

Type `Certificate` into the Windows search bar, click the `Manage Computer Certificates` control panel item that is suggested.

From the Certificate Management program that comes up (certlm), you should now see a `localhost` key under `Personal >> Certificates` .

I copied this certificate into `Trusted Root Certification Authorities` . I'll be honest in that I'm not sure if that's necessary.

Selecting the newly copied certificate, double click on it (the localhost certificate). From the Certificate modal, click the `Details` tab, then the `Copy to File...` button.

This will bring up and Export Wizard, I chose to export the private key, click next. I also chose to `Export all extended properties` (again, I'm not certain if that was necessary). I chose to use a simple password (`pass`) and the default encryption. Choose a folder to export to and name the file. You can always move and rename the file if necessary. For simplicity's sake let's copy it to

Step 3 - Convert `.pfx` file for use with Apache

From here I basically followed the tutorial here, but I'll add instructions here (tweaked for our settings) in case that site goes down.

Open your Command Prompt in the `/apache/conf/` folder
Run the following commands: ***Note:*** *This assumes you have `openssl.exe` in the `bin` folder in the apache root folder (this should be standard/default)*

```
..\bin\openssl pkcs12 -in myCert.pfx -nocerts -out privateKey.pem
```

This will prompt you for a password, enter what you input for Step 2 when you exported the `.pfx` file. In my case, this is `pass`. I entered the same password for the PEM phrase and again to verify. This will create a new file called `privateKey.pem` in your conf folder.

Then, run

```
..\bin\openssl rsa -in privateKey.pem -out private.pem
```

Again you will be prompted for a password (`Enter pass phrase for privateKey.pem:`), use the password you set for `privateKey.pem`. (In my case, `pass`)
You should see a message that says `writing RSA key` and a new file called `private.pem` in your `conf/` folder. **This will be your SSLCertificateKeyFile.**

Now to generate the corresponding Server Certificate. Run:

```
..\bin\openssl pkcs12 -in myCert.pfx -clcerts -nokeys -out EntrustCert.pem
```

This will prompt you for a password, enter what you input for Step 2 when you exported the `.pfx` file. Enter it and you will now have a file called `EntrustCert.pem` in your `conf` folder. **This is your SSLCertificateFile**

Step 4 - Configure `httpd.conf`

Use the new files created as you server's key and certificate. Be sure to change your document root to where your files are!

```
ServerName localhost:80
Protocols h2 h2c http/1.1
<Directory />
    Options FollowSymLinks
    AllowOverride All
</Directory>
```

```
    SSLEngine on
    SSLCertificateFile "${SRVROOT}/conf/EntrustCert.pem"
    SSLCertificateKeyFile "${SRVROOT}/conf/private.pem"
  </VirtualHost>
```

Also in `httpd.conf` :

- Make sure `LoadModule ssl_module modules/mod_ssl.so` is uncommented (no `#` in front)
- Uncomment `LoadModule socache_shmcb_module modules/mod_socache_shmcb.so`
- Uncomment `LoadModule http2_module modules/mod_http2.so`
- Uncomment `Include conf/extra/httpd-ssl.conf` *(NOTE: Ensure that's where the file is!)*

I also have curl and open ssl libraries included:

```
# load curl and open ssl libraries
LoadFile "C:\php\libeay32.dll"
LoadFile "C:\php\ssleay32.dll"
LoadFile "C:\php\libssh2.dll"
```

These modules shouldn't be necessary, but I'll note that I have them enabled:

```
LoadModule rewrite_module modules/mod_rewrite.so
LoadModule filter_module modules/mod_filter.so
LoadModule deflate_module modules/mod_deflate.so
```

Step 5 - Config `httpd-ssl.conf`

In the `extra/` folder in the `conf/` folder you should see a file called `httpd-ssl.conf` .

5a. Change the `DocumentRoot` - Change the `DocumentRoot` from the default to the directory where your files are.

5b. Change the `ServerName` - Change the `ServerName` from the default (something like `www.example.com:443`) to `localhost:443`

5c. Change the `SSLCertificateFile`
Change the `SSLCertificateFile` from the default (`${SRVROOT}/conf/server.crt`) to `${SRVROOT}/conf/EntrustCert.pem`

5c. Change the `SSLCertificateKeyFile`
Change the `SSLCertificateKeyFile` from the default (`${SRVROOT}/conf/server.key`) to `${SRVROOT}/conf/private.pem`

All together, in the `<VirtualHost _default_:443>` tag.

```
ServerAdmin admin@example.com
ErrorLog "${SRVROOT}/logs/error.log"
TransferLog "${SRVROOT}/logs/access.log"

#   SSL Engine Switch:
#   Enable/Disable SSL for this virtual host.
SSLEngine on

#   Server Certificate:
#   Point SSLCertificateFile at a PEM encoded certificate.  If
#   the certificate is encrypted, then you will be prompted for a
#   pass phrase.  Note that a kill -HUP will prompt again.  Keep
#   in mind that if you have both an RSA and a DSA certificate you
#   can configure both in parallel (to also allow the use of DSA
#   ciphers, etc.)
#   Some ECC cipher suites (http://www.ietf.org/rfc/rfc4492.txt)
#   require an ECC certificate which can also be configured in
#   parallel.
SSLCertificateFile "${SRVROOT}/conf/EntrustCert.pem"
#SSLCertificateFile "${SRVROOT}/conf/server-dsa.crt"
#SSLCertificateFile "${SRVROOT}/conf/server-ecc.crt"

#   Server Private Key:
#   If the key is not combined with the certificate, use this
#   directive to point at the key file.  Keep in mind that if
#   you've both a RSA and a DSA private key you can configure
#   both in parallel (to also allow the use of DSA ciphers, etc.)
#   ECC keys, when in use, can also be configured in parallel
SSLCertificateKeyFile "${SRVROOT}/conf/private.pem"
#SSLCertificateKeyFile "${SRVROOT}/conf/server-dsa.key"
#SSLCertificateKeyFile "${SRVROOT}/conf/server-ecc.key"
```

Restart Apache

After making these changes you should be able to restart Apache and navigate to https://localhost without a security warning and a little padlock!

I hope this helps someone! ☺

Sources:

1.) Auri Rahimzadeh's answer on creating a self-signed certificate

2.) Entrust Datacard - How do I convert a .pfx to be used with an Apache server?

Share Improve this answer Follow

edited Mar 19 '19 at 18:53

answered Mar 19 '19 at 18:40

StephanieQ

742 1 8 20

1 Thank you--worked for me using Windows 10, Apache24. Firefox warned that my certificate was self-signed but after I proceeded anyway, the lock icon is marked with a warning flag stating that I have granted an exception to it. – Dale Thompson Jul 26 '20 at 20:34

1

This HowTo for CentOS was easy to follow and only took about 5 minutes: https://wiki.centos.org/HowTos/Https

I won't detail each step here, but the main steps are:

1.) Install the openssl module for apache, if not already installed

2.) Generate a self-signed certificate

--At this point, you should be able to visit https://localhost successfully

3.) Set up a virtual host if needed

Share Improve this answer Follow

answered Apr 11 '17 at 18:33

KayakinKoder
2,222 1 15 27

1

This worked on Windows 10 with Apache24:

1 - Add this at the bottom of `C:/Apache24/conf/httpd.conf`

```
Listen 443
<VirtualHost *:443>
    DocumentRoot "C:/Apache24/htdocs"
    ServerName localhost
    SSLEngine on
    SSLCertificateFile "C:/Apache24/conf/ssl/server.crt"
    SSLCertificateKeyFile "C:/Apache24/conf/ssl/server.key"
</VirtualHost>
```

2 - Add the `server.crt` and `server.key` files in the `C:/Apache24/conf/ssl` folder. See other answers on this page to find those 2 files.

That's it!

Share Improve this answer Follow

answered Apr 13 '18 at 18:10

jogi99
399 1 2 10

Yes this works. I used step 1 - 3 from StephanieQ to generate the certificate and than "openssl x509 -req -in server.csr -signkey server.key -out server.crt" to generate the *.crt file with cygwin. – b3wii Apr 8 '19 at 14:31

I don't have an `httpd.conf` , what file do I put it in? Here's what happens when I put this text inside `apache2.conf` : `Invalid command 'SSLEngine', perhaps misspelled or defined by a module not included in the server configuration` – Aaron Franke Jul 13 '20 at 1:01

1

```
ssh -R youruniquesubdomain:80:localhost:3000 serveo.net
```

And your local environment can be accessed from https://youruniquesubdomain.serveo.net

Serveo is the best

- No signup.
- No install.
- Has HTTPS.
- Accessible world-wide.
- You can specify a custom fix, subdomain.
- You can self host it, so you can use your own domain, and be future proof, even if the service goes down.

I couldn't believe when I found this service. It offers everything and it is the easiest to use. If there would be such an easy and painless tool for every problem...

Share Improve this answer Follow edited Jul 11 '18 at 13:16

answered Jul 11 '18 at 13:11

totymedli
22.2k 18 106 140

This doesn't work anymore in 2020, the page loads forever. However, for some reason the SSH command still works...? – Aaron Franke Jul 13 '20 at 0:53

1

I'd like to add something to the very good answer of @CodeWarrior, that works perfectly on Chrome, but for Firefox needs an additional step.

Since Firefox does not thrust CA Certificates that Windows does by default, you need to go on `about:config` , scroll down to `security.enterprise_roots.enabled` and change it to true.

Now your certificate should be seen as valid also on Firefox.

Of course this is only for development purposes, since ssl trust is a critical security concern and change this settings only if you know the implications.

Share Improve this answer Follow

answered Jun 30 '19 at 3:59

RDev
946 1 9 17

0

```
openssl req -new -x509 -keyout server.pem -out server.pem -days 365 -nodes
```

Note: Assuming you have openssl installed.

2. Save below code in a file named `simple-https-server.py` in **any** directory you want to run the server.

```
import BaseHTTPServer, SimpleHTTPServer
import ssl

httpd = BaseHTTPServer.HTTPServer(('localhost', 4443),
SimpleHTTPServer.SimpleHTTPRequestHandler)
httpd.socket = ssl.wrap_socket (httpd.socket, certfile='./server.pem', server_side=True)
httpd.serve_forever()
```

3. Run the server from terminal:

```
python simple-https-server.py
```

4. Visit the page at:

```
https://localhost:4443
```

Extra notes::

1. You can change the **port** in `simple-https-server.py` file in line

```
httpd = BaseHTTPServer.HTTPServer(('localhost', 4443),
SimpleHTTPServer.SimpleHTTPRequestHandler)
```

2. You can **change `localhost` to your IP** in the same line above:

```
httpd = BaseHTTPServer.HTTPServer(('10.7.1.3', 4443),
SimpleHTTPServer.SimpleHTTPRequestHandler)
```

and access the page on any device your network connected. This is very handy in cases like "you have to test HTML5 GeoLocation API in a mobile, and Chrome restricts the API in secure connections only".

Gist: https://gist.github.com/dergachev/7028596

http://www.piware.de/2011/01/creating-an-https-server-in-python/

Share Improve this answer Follow

answered Jan 11 '17 at 5:27

Asim K T
11.9k 7 64 84

0

For those using macOS this is a great guide https://getgrav.org/blog/macos-sierra-apache-multiple-php-versions to set up your local web dev environment. In its 3rd part https://getgrav.org/blog/macos-sierra-apache-ssl Andy Miller explains how to set up apache with

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout server.key -out server.crt
```

But there are a few steps you need to follow, so check that out and good luck! ;)

Share Improve this answer Follow

answered Dec 7 '17 at 11:01

tiomno

1,753 16 27

Highly active question. Earn 10 reputation in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.