4.4 Apache and SSL Certificates

Introduction

> This is really long so consider breaking it up with a summary of steps in the beginning. Also, how can it be made more succinct? Perhaps an abridged version needs to be created.

This tutorial shows you how to setup Apache with a new SSL Certificates for web sites. Please read Apache - Renewing SSL Certificates for the renewal process.

Select SSL Certificate

SSLShopper provides an overview of the different types of SSL certificates available with pros and cons. If you are still unsure, use the SSL Shopper Wizard to guide you.

Sample Selection

We used the SSL Shopper Wizard with the following criteria,

- Secure one domain name or network name
- Just need it to be secure with no warning messages
- Price range per year $0-$100

From the results, we chose a **free** simple SSL certificate from the CA StartCom in a SSL certificate package called StartSSL Free.

Note there is a newer service (checked July 2018), a non-profit called Let's Encrypt that provides free SSL certificates. To understand what you get, you may look at their Hello World site.

SSL Setup (using openssl)

Server keys must be generated for the Certificate Signing Request (CSR). Openssl will be used to generate this CSR.

Generate Server Private Key

There are two options for generating the key,

1. Without Passphrase Encryption
2. With Passphrase Encryption

Because passphrase encryption requires an administrator's intervention, the standard is to **not** use passphrase encryption and rely on the file system to protect the keys.

> We will continue with the BonsaiFramework example and be creating an SSL certificate which will work for both https://www.earth.com and https://earth.com.

Without Passphrase Encryption

As described in the Security Configuration Benchmark for Apache HTTP Server 2.2 v3.0.0, the generally accepted method of generating the key is without passphrase encryption,

```
su bhitch # Use a sudo enabled account.
cd ~
mkdir private
chmod -R 700 ./private
cd private
openssl genrsa -out www.earth.com_server.key 2048
```

The openssl command reads,

- 2048 - make the RSA private key 2048 bit
- The default file format will be PEM
- The default encoding of the file will be base64

> Notice the creation of the private directory. It is very important that only the proper administrators should have access to the private key.

Generate the CSR and Public Key

Generate the CSR (Certificate Signing Request) which will be submitted to the CA using the private key **www.earth.com_server.key**.

```
openssl req -new -key www.earth.com_server.key -out
www.earth.com_server.csr
```

You will be prompted to enter information about the certificate. The values should reflect your organization.

A not so obvious prompt is **Common Name (eg, YOUR Name)**. This value should not be your name. Instead it should be the domain name of your website. In our example, it could be **earth.com** or **www.earth.com**.

We want to use both www in our domain name. Also we prefer using www, so enter, **www.earth.com**.

> CA's such as StartSSL can have SSL support for both www and no www. This is possible because the CA issues certificates where the **Certificate Subject Alt Name** extension is populated by both DNS Names, **www.earth.com** and **earth.com**.
>
> To make it work, **make sure to use www in the Common Name**.

Prompts from Running openssl req

```
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]:CA
State or Province Name (full name) [Some-State]:Ontario
Locality Name (eg, city) []:Toronto
Organization Name (eg, company) [Internet Widgits Pty Ltd]:The Planet Earth
Incorporated
Organizational Unit Name (eg, section) []:Earth Defence
Common Name (eg, YOUR name) []:www.earth.com
Email Address []:admin@earth.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

The CSR file will be used by the CA to issue the Web Server's certificate.

Here are some useful details about the CSR,

- The default is a PEM Base64 encoded format.
- The private key used to digitally sign the CSR.
- The CSR command will also generate the public key and store it within the CSR file.

Optionally verify your CSR via the command line or submit the contents to the SSLShopper CSR Decoder,

```
openssl req -in www.earth.com_server.csr -noout -text
```

Submit Public Key to CA

The CSR file is submitted to the CA. Every CA will have a slightly different procedure.

Help Improve the Article
Add the exact steps for StartSSL here as a reference.

Domain Validation requires an email that is listed in the Domain Management Administrative and Technical contact make sure they are using a real email or if not change it to a real email and then change it back

This step also includes various CA specific procedures to prove your identity.
You may also be prompted for a host name underneath your domain. If you want to use both https://www.earth.com and https://earth.com then make sure both are listed here.

Download Certificates

The CA will use your public key in the CSR to sign and return back your **server certificate**.

Download Signed Server Certificate

Following this example using StartSSL we are provided an email to follow instructions to retrieve the certificate through their web interface,

> Toolbox, Retrieve Certificate, select your certificate. The result is often a PEM base64 encoded certificate which will be a file or webpage. In our case it is a webpage where you must copy and paste the Certificate text data into a text file.

The text file contains your server public key signed by the CA. Save the file with the domain name and extensions to denote encoding and type. In this example, the file name will be,

```
www.earth.com_server.signed_cert.crt
```

Verifying Signed Server Certificate

section to be completed by Roderick
To verify your signed server certificate in Windows change the certificate file extension to **crt**. Then just double-click the file and you should at minimum confirm,

1. The DN (Distinguished Name) matches your existing request.
2. Bit size of the certificate matches your request.
3. Expiry date of the certificate and record that in a calender to remind you for renewal.

ADD IMAGE

...

Q: I did not register the certificates, someone just sent me a bunch of files and I do not know what is what.

A: Read the article, Certificate File Formats which explains the file types and also how to verify and validate certificate files.

Download Chain Certificate(s)

Almost all CAs require the server install **one** or **more** chain certificates.

You may download the chain certificate from your CA's website (you may need to search for it) or more conveniently download from SSLShopper's list of Chain Certificates and respective CA Installation Instructions

> Chain Certificates also expire. Whenever, you update your SSL certificate, you will also want to check if a new chain certificate is available.

In this case the chain certificate was found at StartCom How to Install Apache Server webpage,

- ca.pem is the root CA certificate
- sub.class1.server.ca.pem is the chain certificate

The file names, extension and documentation on file formats for certificates is a mess. As explained in Certificate File Formats we can identify the file as PEM and base64 encoded. There is no real standard that everyone follows so we will rename the file using the BonsaiFramework standards,

```
mv sub.class1.server.ca.pem sub.class1.server.ca.crt
```

After the extension change double click on the file in Windows to inspect the certificate.

Copy the **Issued to**: name as this will be used to properly name the file,

```
mv sub.class1.server.ca.pem
StartCom_Class_1_Primary_Intermediate_Server_CA.crt
```

Concatenating Chain Certficates

> Put example of Verisign here. Note that order matters and how to determine this if the CA did not provide it concatenated.

What About the CA Certificate?

You don't need it. Yes, it is noted in many tutorials on the Internet, but as explained in the Apache 2 documentation,

> *These are used to verify the client certificate on Client Authentication.*

In other words, you only need CA certificates on Web Servers if you intend to have the Browsers authenticate and identify themselves.

Store Certificates

In Ubuntu, the default location for SSL certificates are,

```
/etc/ssl/private/ # Only view-able by root the standard location for the private keys
/etc/ssl/certs/ # Standard location for CA keys with symbolic links pointing to /usr/share/ca-certificates/
```

For now we will use this structure.

> This needs some consideration of structure and permissions.
>
> However, are the issues with using the default Ubuntu locations.
>
> - Putting chain certificates in the same directory as CA certificates does not make sense if SSLCACertificatePath is used.
> - Developing a portable BonsaiFramework version of Apache Web Server and keeping certificates with the service makes more sense.
> - One other alternative is to use Debian's apparent standard, /etc/apache/ssl but I noticed there is **no** consideration for protecting the private key.

Store Public Key

Store the **public key** in the public folder,

```
sudo cp www.earth.com_server.signed_cert.crt /etc/ssl/certs
sudo chown root:root /etc/ssl/certs/www.earth.com_server.signed_cert.crt
```

> The above chown root:root command ensure the signed public key is protected. Also, if you are using a user other than root to start Apache, then adjust the file ownership to that user.

Storing Chain Certificates

Chain certificates can be stored in the same location as the public certificates,

```
sudo cp StartCom_Class_1_Primary_Intermediate_Server_CA.crt /etc/ssl/certs/
sudo chown root:root
/etc/ssl/certs/StartCom_Class_1_Primary_Intermediate_Server_CA.crt
```

You may notice that Ubuntu itself might already have a matching chain certificate under a different file name. This chain certificate can be used instead and it may be automatically updated by the Ubuntu Operating System (however, I have not found any documentation about how this actually works).

Storing Private Keys

Ubuntu has a pre-configured location for private keys, **/etc/ssl/private**.

Notice the permissions on the /etc/ssl/private folder is 710 and owned by root and the group ssl-cert,

```
ls -al /etc/ssl/
total 44
drwxr-xr-x  4 root root      4096 2011-04-07 10:15 .
drwxr-xr-x 71 root root      4096 2011-06-08 14:22 ..
drwxr-xr-x  2 root root     20480 2011-06-21 11:41 certs
-rw-r--r--  1 root root      9374 2010-10-06 20:51 openssl.cnf
drwx--x---  2 root ssl-cert  4096 2011-06-13 20:59 private
```

If you opt to use your own private folder makes sure to set the same permissions as /etc/ssl/private/

```
sudo chown -R root:ssl-cert /opt/apache/httpd/ssl/private/* # Make
the user starting Apache the owner, in this case it is root.
sudo chmod 710 /opt/apache/httpd/ssl/private/
```

Again, I will stress that this is very important!

Store the **private key** into the protected folder. This process is derived from the Security Configuration Benchmark Apache HTTP Server 2.2 Version 3.0.0 p60.

```
sudo cp www.earth.com_server.key /etc/ssl/private/
sudo chown root:ssl-cert /etc/ssl/private/www.earth.com_server.key # Ensure
the file is owned by the right users.
sudo chmod 640 /etc/ssl/private/www.earth.com_server.key # Secure the file
```

The ssl-cert group is a special group to make it easy for other processes to use certs.

Why does the server key need both read and write (640) for the owner?

I'm not sure actually, but I have found that not giving write permission appeared to result in me not being able to properly reload Apache.

Honestly, there were other factors so I'm' not sure if write permission actually solved my issue. It worked, and I have not had the time to investigate this thoroughly. Let me know if you find otherwise.

Next, Apache needs to be setup.

Apache Setup

Enable the Apache Module

Be default the Apache SSL module is not enabled,

```
sudo a2enmod ssl
Enabling module ssl.
See /usr/share/doc/apache2.2-common/README.Debian.gz on how to configure
SSL and create self-signed certificates.
Run '/etc/init.d/apache2 restart' to activate new configuration!
sudo /etc/init.d/apache2 restart
```

Create an SSL Virtual Host

Following along the BonsaiFramework tutorial, we are using virtual hosts.

We'll create an SSL Vitual Host version of www.earth.com. Note that both http://www.earth.com on port 80 and https://www.earth.com on port 443 can co-exist.

> All the SSL entries are derived from etc/apache2/sites-available/default-ssl. Read the comments in that file for a more in depth understanding of the configurations.

First we create the file,

```
cd /etc/apache2/sites-available
sudo cp default-ssl www.earth.com-ssl
```

Building upon the work in BonsaiFramework Apache Virtual Hosting, below are the minimal recommend lines to enable SSL.

```
<IfModule mod_ssl.c>
<VirtualHost *:443>
    ServerAdmin webmaster@localhost

    ServerName www.earth.com
    ServerAlias earth.com

    DocumentRoot /home/www.earth.com/www
    <Directory />
        # This prevents use of .htaccess
        AllowOverride None
    </Directory>

    ErrorLog /var/log/apache2/ssl_www.earth.com.error.log

    # Possible values include: debug, info, notice, warn, error, crit,
    # alert, emerg.
    LogLevel warn

    CustomLog /var/log/apache2/ssl_www.earth.com.access.log combined

    # ----------------------
    # Start Enable SSL
    # -----------------

    # SSL Engine Switch:
```

```
    # Enable/Disable SSL for this virtual host.
    SSLEngine on

    # Load the keys signed key
    SSLCertificateFile /etc/ssl/certs/www.earth.com_server.signed_cert.crt

    # Load the private key
    SSLCertificateKeyFile /etc/ssl/private/www.earth.com_server.key

    # Load the Certificate chain
    SSLCertificateChainFile
/etc/ssl/certs/StartCom_Class_1_Primary_Intermediate_Server_CA.crt

    # Loads all Certificate Authorities in the provided path
    # SSLCACertificatePath /etc/ssl/certs/

    # SSL Engine Options
    <FilesMatch "\.(cgi|shtml|phtml|php)$">
        SSLOptions +StdEnvVars
    </FilesMatch>
    <Directory /usr/lib/cgi-bin>
        SSLOptions +StdEnvVars
    </Directory>

    # SSL Protocol Adjustments
    BrowserMatch "MSIE [2-6]" \
        nokeepalive ssl-unclean-shutdown \
        downgrade-1.0 force-response-1.0
    # MSIE 7 and newer should be able to use keepalive
    # "MSIE [17-9]" matches MSIE 7 to 9 and 10 to 19 (and 1, but that
should not be a problem)
    BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown

    # -----------------
    # End Enable SSL
    # ---------------------
```

```
</VirtualHost>
</IfModule>
```

Line 35 - SSLCertificateChainFile with most modern CAs is absolutely necessary. It is on the onus of the server to provide this and also to keep it up to date.

Line 39 & 40 - Note the commented out lines, I have recently learned that SSLCACertificatePath is another way of loading the certificates by path the method we use is more specific but there are pros and cons that could be added in the future.

> It is not possible to run multiple SSL-enabled virtual hosts on a server with only one IP address. A separate IP address or port is necessary for each SSL-enabled domain. There are new modules that provide this functionality, but as of May 2011 it is not yet widely supported by browsers.

Verify SSL Certificate

Using Browser

Launch a browser and try both https://www.earth.com and https://earth.com.

A lock icon should appear somewhere on your browser to indicate that the browser session is now encrypted.

Certificate details can also be verified by clicking on the lock icon and selecting **View certificates**.

If the website is available on the Internet you can skip to the next section which uses a CA provided SSL Check Service.

If not, make sure perform the browser check with multiple browsers. In particular chain certificates often work on Firefox but not on IE6.

Using CA Provides SSL Check Service

CAs and related services often provide services to verify SSL certs are properly installed on your web server. The only criteria is that your web server must be available on the Internet. Here are a list of services,

- SSLShopper
- VeriSign
- Entrust - private service only available after you retrieve a certificate

> On the topic of having SSL support for both https://www.earth.com and http://earth.com, when inspecting the certificate, navigate to,
>
> 1. www.earth.com
> 2. Certificate
> 3. Extensions
> 4. Certificate Subject Alt Name
>
> Under the Certificate Subject Alt Name will see both DNS entries,
>
> ```
> Not Critical
> DNS Name: www.earth.com
> DNS Name: earth.com
> ```

Checking for Mixed Content

If you inline load images without using a relative path you will get mixed content which makes your page insecure. Tools to check this,

- https://www.jitbit.com/sslcheck/
- https://developers.google.com/web/tools/lighthouse/audits/mixed-content
- https://www.whynopadlock.com/check.php

And here's an example page of common mixed content errors.

Clean Up

Some CSR requests may be re-used to renew the Signed SSL Certificate. However, often most CA's will by process ask for a new CSR even if the original may be reused.

Check with your CA to see how the CSR renewal process works. If the requirement is for a new csr, to avoid confusion it is best to delete the csr request once everything is proven to be working,

```
rm www.earth.com_server.csr
```

Last, it should be noted that the SSL Certificate will expire. Some CA's will use the submitted contact information to notify by email.

> This section can be improved by explaining, how to check the expiry of a certificate using the browser, linked to an article about certificate renewal. Also should note that during renewal it is a good idea to download new chain certs as they also expire.

FAQ

> Expand this section with topics like, are server certs bound to the server?

Are server certs bound to the server?

No, server certs are not bound to the server. You can simply copy/move certs around between servers. In a load balanced environment you would be using the same certs on the different web servers. In a backup scenario you can use the same certificates.

I keep getting this error after I setup SSL for my domain on the second server how do I fix it?

[Thu Jan 26 19:13:25 2012] [warn] RSA server certificate CommonName (CN) `www.domain.com' does NOT match server name!?

Check that you have not enabled the virtual host SSL instead of the just the virtual host because what is happening is with the virtual host SSL enabled its matching www.domain.com-ssl agains the common name www.domain.com.

...

References

https://help.ubuntu.com/10.04/serverguide/C/httpd.html#https-configuration - trying this one first.

http://doc.opensuse.org/products/opensuse/openSUSE/opensuse-reference/cha.apache2.html#sec.apache2.ssl - quickly read and seems to be a good read.

http://www.entrust.net/ssl-technical/webserver.cfm - Entrust provides some decent docs.

http://forums11.itrc.hp.com/service/forums/questionanswer.do?admit=109447626+1304737120524+28353475&threadId=1398455 - good explanation about what is actually happening with the openssl genrsa command.

http://forums.freebsd.org/showthread.php?t=6490 - straight to the point list of commands.

http://allben.net/post/2009/02/01/SSL-Certificate-for-WWW-and-no-WWW.aspx - discussion on www or no www in common name while generating CSR.

http://jasoncodes.com/posts/startssl-free-ssl - complete tutorial, the notes look good and clear.

http://www.ehow.com/how_7811607_create-verified-ssl-certificate.html - alright, I should comment on this one.

http://blog.ruilopes.com/post/3678866680/from-http-to-https-with-free-certificates - consider importing this article's steps on checking if Ubuntu ships with the certificates.