HOW TO GENERATE A CERTIFICATE SIGNING REQUEST (CSR) WITH OPENSSL

Introduction

A Certificate Signing Request (CSR) is the first step in setting up an SSL Certificate on your website. SSL certificates are provided by Certificate Authorities (CA), which require a Certificate Signing Request (CSR).

This guide will instruct you on how to generate a Certificate Signing Request using OpenSSL.

Prerequisites

- Access to a user account with root or **sudo** privileges
- A command line/terminal window
- If you're working on a remote server, an established SSH connection to the server
- OpenSSL needs to be installed on your system to generate the key
- A text editor, such as **nano**, to view your key

GENERATE A OPENSSL CERTIFICATE SIGNING REQUEST

Step 1: Log into Your Server

Open a terminal window. Use your SSH connection to log into your remote server.

Note: If you are working locally, you don't need an SSH connection. Also, most Linux systems will launch a terminal window by pressing Ctrl-Alt-T or Ctrl-Alt-F1.

Step 2: Create an RSA Private Key and CSR

It is advised to issue a new private key each time you generate a CSR. Hence, the steps below instruct on how to generate both the private key and the CSR.

```
openssl req -new -newkey rsa:2048 -nodes -keyout your_domain.key -out your_domain.csr
```

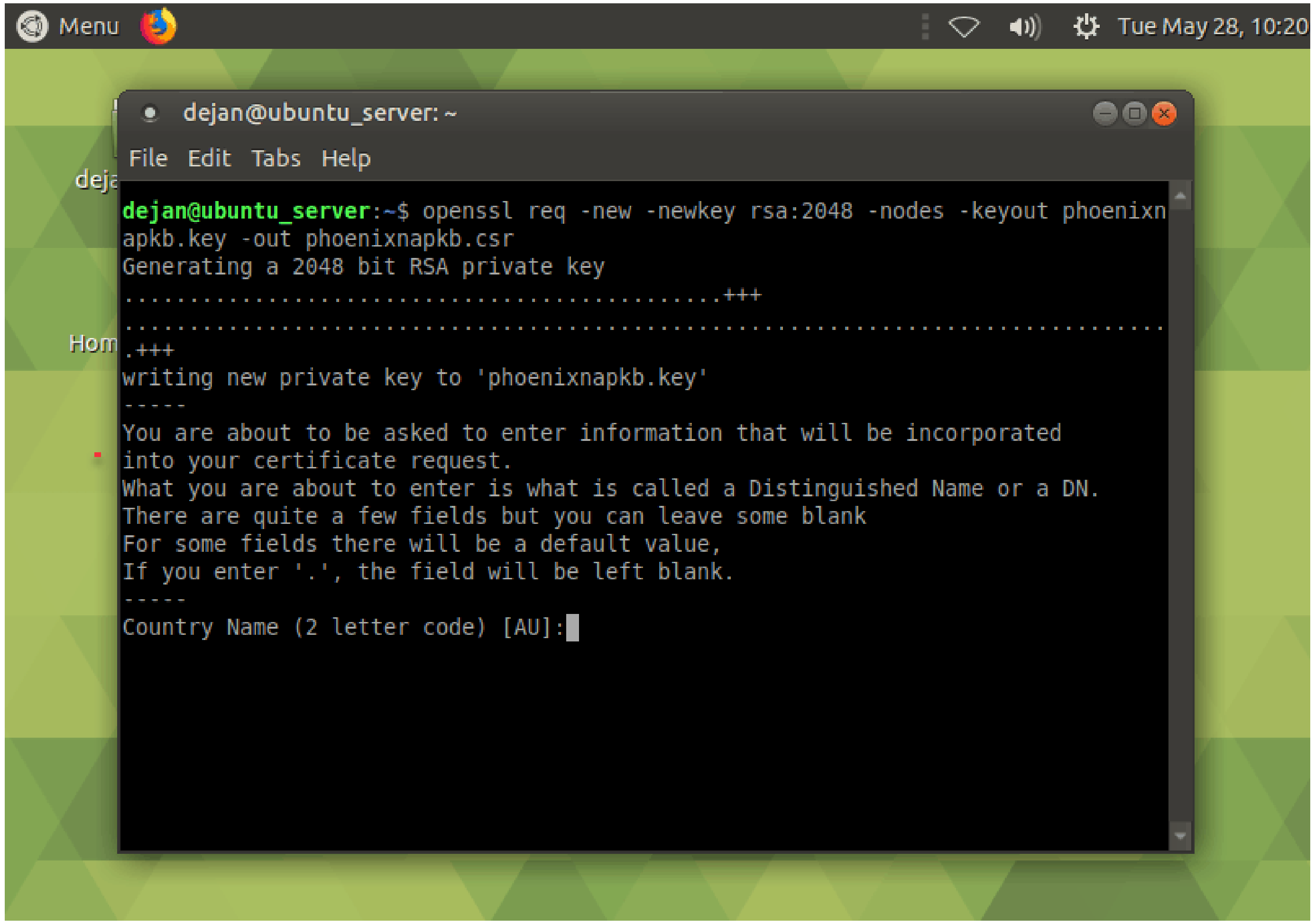
Make sure to replace `your_domain` with the actual domain you're generating a CSR for.

The commands are broken out as follows:

- **openssl** – activates the OpenSSL software
- **req** – indicates that we want a CSR
- **–new –newkey** – generate a new key
- **rsa:2048** – generate a 2048-bit RSA mathematical key
- **–nodes** – no DES, meaning do not encrypt the private key in a PKCS#12 file
- **–keyout** – indicates the domain you're generating a key for

- **–out** – specifies the name of the file your CSR will be saved as

Note: Use 2048-bit key pairs. The 4096-bit key pairs are more secure, however, they require a lot more server resources.



```
dejan@ubuntu_server:~$ openssl req -new -newkey rsa:2048 -nodes -keyout phoenixn
apkb.key -out phoenixnapkb.csr
Generating a 2048 bit RSA private key
.............................................+++
................................................................................
.+++
writing new private key to 'phoenixnapkb.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) [AU]:
```

Step 3: Enter Your CSR Information

Your system should launch a text-based questionnaire for you to fill out.

Enter your information in the fields as follows:

- **Country Name** – use a 2-letter country code (US for the United States)
- **State** – the state in which the domain owner is incorporated
- **Locality** – the city in which the domain owner is incorporated
- **Organization name** – the legal entity that owns the domain
- **Organizational unit name** – the name of the department or group in your organization that deals with certificates
- **Common name** – typically the fully qualified domain name (FQDN), i.e. what the users type in a web browser to navigate to your website
- **Email address** – the webmaster's email address
- **Challenge password** – an optional password for your key pair

Please take into account that *Organization Name* and *Unit Name* must not contain the following characters:

```
< > ~ ! @ # $ % ^ * / \ ( ) ? . , &
```

Step 4: Locate Certificate Signing Request File

Once the software finishes, you should be able to find the CSR file in your working directory.

You can also enter the following:

```
ls *.csr
```

The system should list out all certificate signing requests on the system. The one that matches the domain name you provided in Step 2 appended with the .csr extension is the one you need to look into.

Step 5: Submit the CSR as Part of Your SSL Request

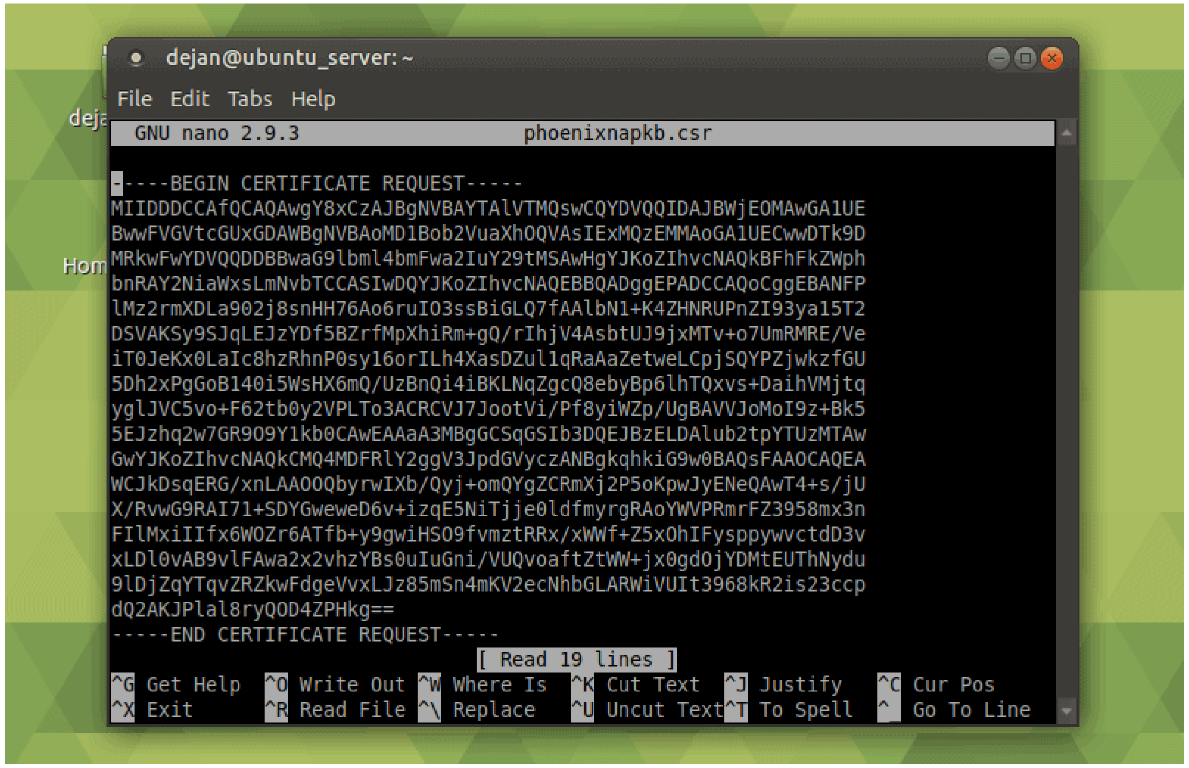
You can open the .csr file in a text editor to find the alphanumeric code that was generated.

Enter the following command:

```
sudo nano your_domain.csr
```

This text can be copied and pasted into a submittal form to request your SSL certificate from a Certificate Authority.

Make sure you copy the entire text. Some CAs may allow you to simply upload the .csr file you generated. Below is an example of a CSR.



```
  GNU nano 2.9.3                       phoenixnapkb.csr

-----BEGIN CERTIFICATE REQUEST-----
MIIDDDCCAfQCAQAwgY8xCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJBWjEOMAwGA1UE
BwwFVGVtcGUxGDAWBgNVBAoMD1Bob2VuaXhOQVAsIExMQzEMMAoGA1UECwwDTk9D
MRkwFwYDVQQDDBBwaG9lbml4bmFwa2IuY29tMSAwHgYJKoZIhvcNAQkBFhFkZWph
bnRAY2NiaWxsLmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANFP
lMz2rmXDLa902j8snHH76Ao6ruIO3ssBiGLQ7fAAlbN1+K4ZHNRUPnZI93ya15T2
DSVAKSy9SJqLEJzYDf5BZrfMpXhiRm+gQ/rIhjV4AsbtUJ9jxMTv+o7UmRMRE/Ve
iT0JeKx0LaIc8hzRhnP0sy16orILh4XasDZul1qRaAaZetweLCpjSQYPZjwkzfGU
5Dh2xPgGoB140i5WsHX6mQ/UzBnQi4iBKLNqZgcQ8ebyBp6lhTQxvs+DaihVMjtq
yglJVC5vo+F62tb0y2VPLTo3ACRCVJ7JootVi/Pf8yiWZp/UgBAVVJoMoI9z+Bk5
5EJzhq2w7GR9O9Y1kb0CAwEAAaA3MBgGCSqGSIb3DQEJBzELDAlub2tpYTUzMTAw
GwYJKoZIhvcNAQkCMQ4MDFRlY2ggV3JpdGVyczANBgkqhkiG9w0BAQsFAAOCAQEA
WCJkDsqERG/xnLAAOOQbyrwIXb/Qyj+omQYgZCRmXj2P5oKpwJyENeQAwT4+s/jU
X/RvwG9RAI71+SDYGweweD6v+izqE5NiTjje0ldfmyrgRAoYWVPRmrFZ3958mx3n
FIlMxiIIfx6WOZr6ATfb+y9gwiHSO9fvmztRRx/xWWf+Z5xOhIFysppywvctdD3v
xLDl0vAB9vlFAwa2x2vhzYBs0uIuGni/VUQvoaftZtWW+jx0gd0jYDMtEUThNydu
9lDjZqYTqvZRZkwFdgeVvxLJz85mSn4mKV2ecNhbGLARWiVUIt3968kR2is23ccp
dQ2AKJPlal8ryQOD4ZPHkg==
-----END CERTIFICATE REQUEST-----
                               [ Read 19 lines ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line
```

You needn’t send the private key to the CA. Once you get your SSL certificate, the private key on the server will bind with it to encrypt communication.

Conclusion

Now you know **how to generate an OpenSSL certificate signing request.** Before submitting the CSR to a certificate authority, we recommend verifying the information it holds. Use one of the widely available online CSR decoders.

SSL is a crucial protocol for securing traffic between a website and its visitors. It helps to protect sensitive information online, such as credit card data.

HOW TO INSTALL SSL CERTIFICATE ON APACHE FOR CENTOS 7

Introduction

SSL Certificates are small data files that certify ownership of a public cryptographic key. Certificate Authorities (CA) guarantee that the key belongs to an organization, server, or other entity listed in the certificate.

When a user, via their browser, accesses a certified website, the information is encrypted with a unique **public key.** The data can only be decrypted by using a unique **private key** located on the host server. This high level of encryption prevents unauthorized attempts to access the information.

In this tutorial, learn **how to install an SSL Certificate on CentOS 7.**

Prerequisites

- A user with **sudo** privileges
- Access to a command line (Ctrl-Alt-T)
- A CentOS 7 machine
- A valid domain name with DNS pointed at the server

How to Get an SSL Certificate

There are several ways to obtain Certificates:

1. Using an automated and free certificate authority such as the Let’s Encrypt project.
2. Commercial certificate authorities provide certificates for a fee (Comodo, DigiCert, GoDaddy)
3. Alternatively, it is possible to create a **self-signed certificate.** This type of certificate is useful for testing purposes or for use in a development environment.

If you are still considering what type of certificate you need, or which CA to choose, we’ve prepared a comprehensive guide to SSL certificates, private keys, and CSRs to assist you in the process.

Note: Trusted CAs **do not verify** self-signed certificates. Users cannot use it to validate the identity of their server automatically.

Install SSL Certificate with Let’s Encrypt

Let’s Encrypt is a free, open, and automated certificate authority. It uses the **certbot** software tool to administer certificates automatically.

Certbot is a highly automated tool. Make sure that that your Apache installation is valid and that you have a virtual host configured for your domain/s. You should first read our tutorial on how to install Apache on CentOS 7 if you need assistance with configuring your firewall and virtual hosts.

Certbot Installation

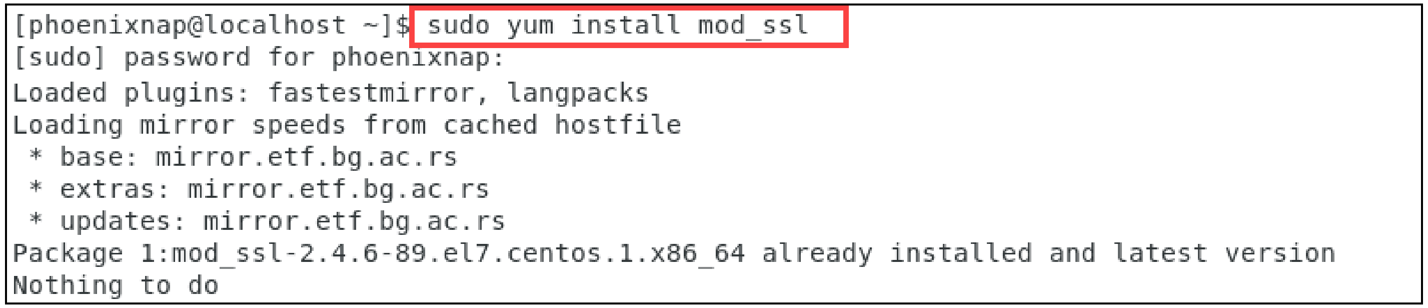
1. Use the command terminal to install the **EPEL** repository and **yum-utils**:

```
sudo yum -y install epel-release yum-utils
```

2. Next, install a module that supports SSL for Apache:

```
sudo yum -y install mod_ssl
```

In this example, the latest version of the module is already available.



```
[phoenixnap@localhost ~]$ sudo yum install mod_ssl
[sudo] password for phoenixnap:
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.etf.bg.ac.rs
 * extras: mirror.etf.bg.ac.rs
 * updates: mirror.etf.bg.ac.rs
Package 1:mod_ssl-2.4.6-89.el7.centos.1.x86_64 already installed and latest version
Nothing to do
```

3. We can now install **certbot** for Apache:

```
sudo yum -y install python-certbot-apache
```

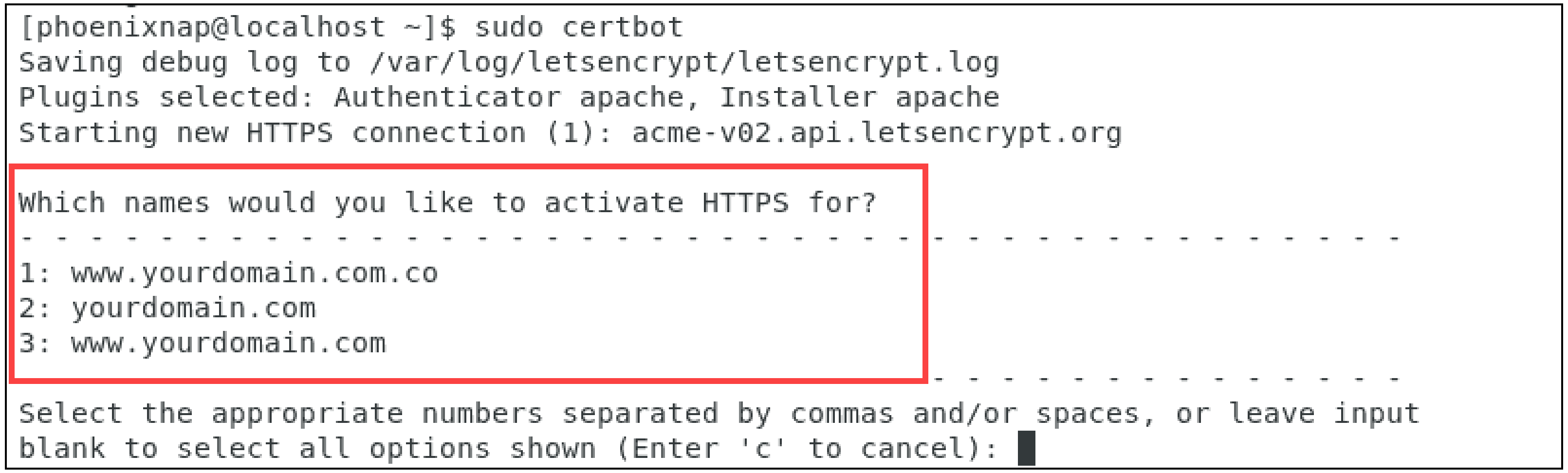
4. Once the installation runs its course, you can start the process to obtain a certificate by entering:

```
sudo certbot -apache -d yourdomain.com
```

Alternatively, start **certbot** by typing:

```
sudo certbot
```

5. The client asks you to provide an email address and to read and accept the Terms of Services. Certbot then lists the domains available on your server. Activate HTTPS for specific domains or all of them by leaving the field blank.



```
[phoenixnap@localhost ~]$ sudo certbot
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator apache, Installer apache
Starting new HTTPS connection (1): acme-v02.api.letsencrypt.org

Which names would you like to activate HTTPS for?
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
1: www.yourdomain.com.co
2: yourdomain.com
3: www.yourdomain.com
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Select the appropriate numbers separated by commas and/or spaces, or leave input
blank to select all options shown (Enter 'c' to cancel):
```

The next prompt allows you to force all requests to secure HTTPS access.

Once you have made your choices, the message on the terminal confirms that you have enabled encryption for your domain.

Automatic Certificate Renewal

The certificates issued by Let's Encrypt are valid for 90 days. The certbot ***renew*** command checks the installed certificates and tries to renew them if they are less than 30 days away from expiration. To automate this process, create a **cron job** to execute the command periodically.

Use your preferred text editor to define how often to execute the **renew** command:

```
sudo crontab –e
```

Enter this line and save the **crontab:**

```
* */12 * * * /usr/bin/certbot renew >/dev/null 2>&1
```

How to Install SSL Certificate with Comodo

1. The first step is to submit a **Certificate Signing Request** to a Certification Authority. Our detailed guide on how to generate a certificate signing request (CSR) with OpenSSL is an excellent resource if you need assistance with this process.

2. Once a CA certifies your request, you receive a copy of your SSL certificate. You can now install the certificate on your CentOS 7 server.

This example shows how to install a certificate from a paid SSL provider, Comodo.

3. Once Comodo verifies your CSR the request, download the SSL files. Copy them (**ComodoRSACA.crt**) and the Primary Certificate (**yourdomain.crt**), to your Apache server directory. The **private key** generated during the CSR (Certificate Signing Request) process needs to be on the same server.

Configure Virtual Hosts for SSL

Aftr you have successfully certified the domain and placed the key files on the server, the next step will be to configure the virtual hosts to display the certificate.

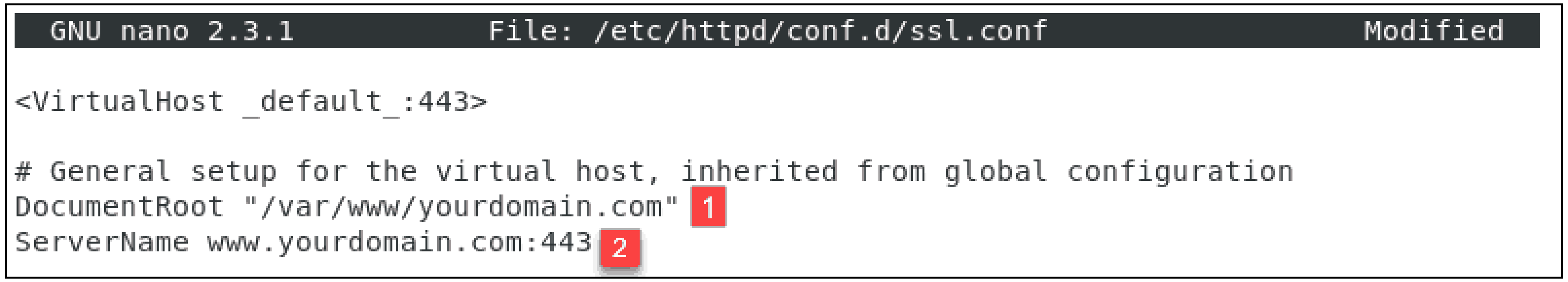
1. Access the SSL configuration file:

```
sudo nano /etc/httpd/conf.d/ssl.conf
```

2. Edit the configuration file to point to the correct files on your server.

Uncomment the following lines under section **<VirtualHost_default_:443>** and enter the correct file paths:

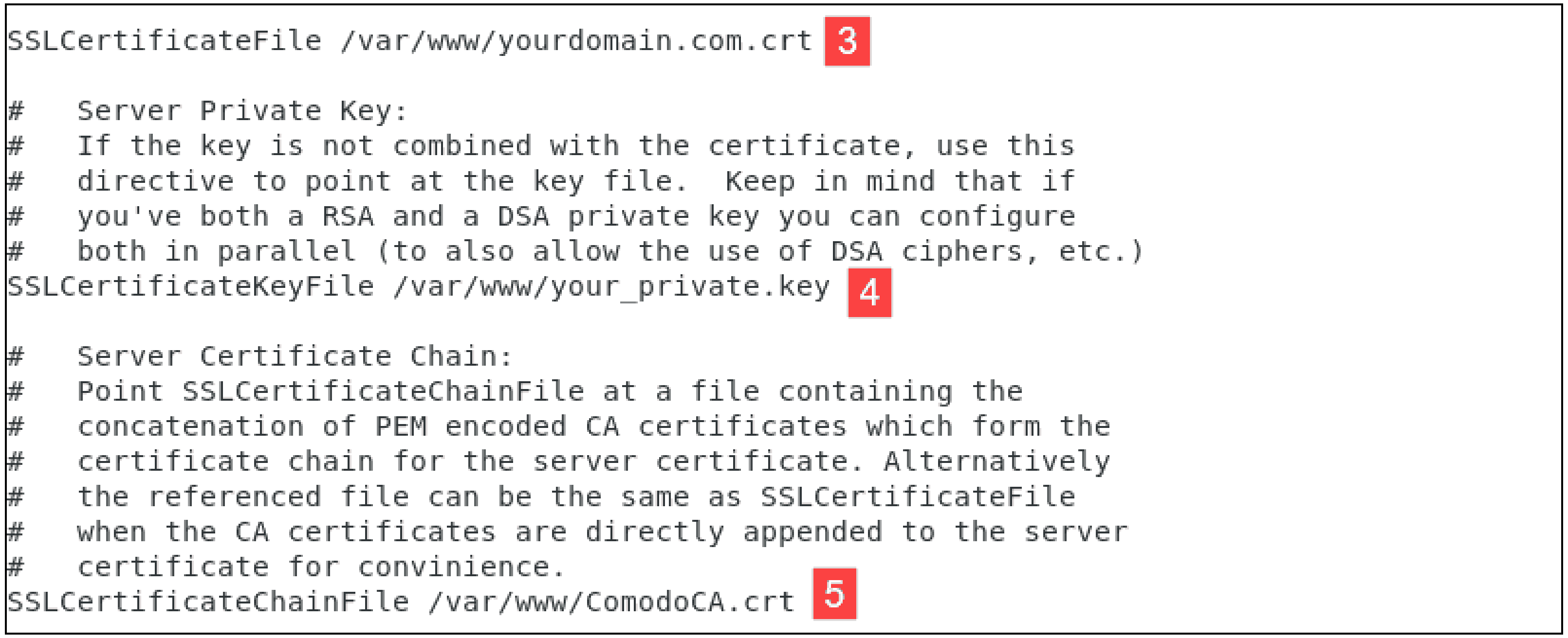
- **DocumentRoot** “/var/www/yourdomain.com”
- **ServerName** yourdomain.com: 443


```
  GNU nano 2.3.1            File: /etc/httpd/conf.d/ssl.conf                    Modified

<VirtualHost _default_:443>

# General setup for the virtual host, inherited from global configuration
DocumentRoot "/var/www/yourdomain.com" 1
ServerName www.yourdomain.com:443 2
```

- **SSLEngine** on
- **SSLCertificateFile** – The path of your certificate file.
- **SSLCertificateKeyFile** – The path of your key file.
- **SSLCertificateChainFile**– The intermediate COMODO certificate file.


```
SSLCertificateFile /var/www/yourdomain.com.crt 3

#   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.)
SSLCertificateKeyFile /var/www/your_private.key 4

#   Server Certificate Chain:
#   Point SSLCertificateChainFile at a file containing the
#   concatenation of PEM encoded CA certificates which form the
#   certificate chain for the server certificate. Alternatively
#   the referenced file can be the same as SSLCertificateFile
#   when the CA certificates are directly appended to the server
#   certificate for convinience.
SSLCertificateChainFile /var/www/ComodoCA.crt 5
```

3. After making the necessary changes, exit the file (Ctrl+X), and press **y** to save the changes.

4. Test your Apache configuration before restarting. Make sure that the syntax is correct by typing:

```
sudo apachectl configtest
```

5. Once the system confirms that the syntax is correct, restart Apache:

```
sudo systemctl restart httpd
```

You have now set up your Apache server to use the SSL certificate.

How to Create a Self-signed SSL Certificate

A self-signed certificate is useful for testing, in development environments, and on an intranet.

1. As with Let's Encrypt, the **mod_ssl** Apache module provides support for the SSL encryption:

```
sudo yum –y install mod_ssl
```

2. Create a new directory to store the private key:

```
sudo mkdir /etc/ssl/privatekey
```

3. Restrict access to that directory only to the root user:

```
sudo chmod 700 /etc/ssl/privatekey
```

4. Generate a self-signed certificate using this OpenSSL command:

```
sudo openssl req -x509 -new -newkey rsa:2048  -nodes -days 365 -keyout /etc/ssl/privatekey/ yourdomain.key -out
/etc/ssl/certs/yourdomain.csr
```

This is a detailed overview of the elements:

- **openssl** – activates the OpenSSL software
- **req** – indicates that we require a CSR
- **-x509** – specifies to use the X.509 signing request
- **-new -newkey** – generate a new key
- **rsa:2048** – generate a 2048-bit RSA mathematical key
- **-nodes** – no DES, meaning do not encrypt the private key in a PKCS#12 file
- **–days 365–** number of days that the certificate is valid for
- **-keyout** – indicates the domain you're generating a key for
- **-out** – specifies the name of the file that contains the CSR

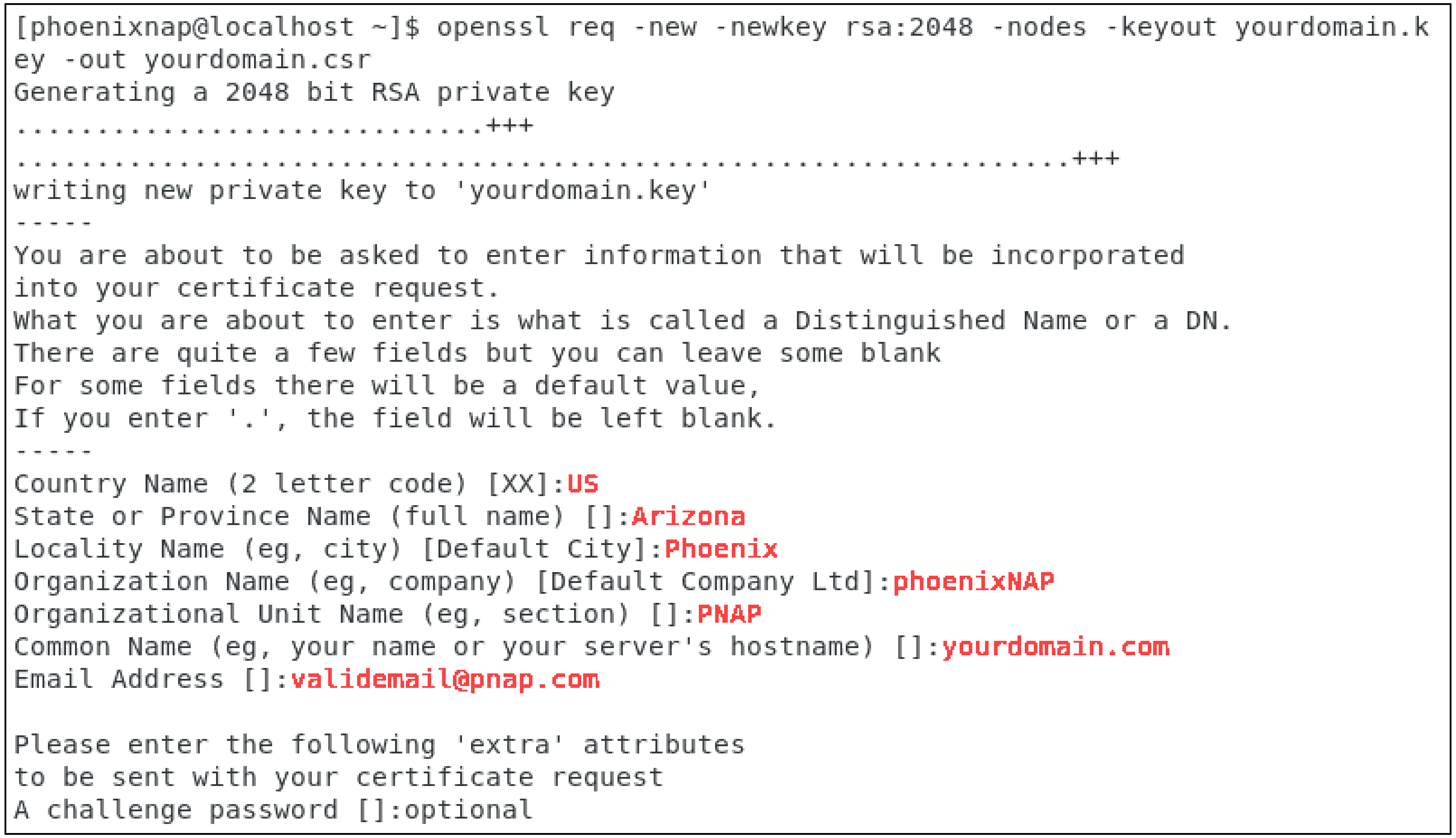
Note: Make sure to replace **yourdomain** with your actual domain.

5. The system launches a questionnaire for you to fill out.

Enter your information in the available fields:

- **Country Name** – use a 2-letter country code
- **State** – the state where the domain owner is incorporated in
- **Locality** – the city where the domain owner is incorporated in
- **Organization name** – an entity that owns the domain
- **Organizational unit name** –the department or group in your organization that works with certificates
- **Common name** – most often, the fully qualified domain name (FQDN)
- **Email address** – contact email address
- **Challenge password** – define an optional password for your key pair

The image represents an example questionnaire in CentOS 7.



```
[phoenixnap@localhost ~]$ openssl req -new -newkey rsa:2048 -nodes -keyout yourdomain.k
ey -out yourdomain.csr
Generating a 2048 bit RSA private key
.............................+++
.................................................................+++
writing new private key to 'yourdomain.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) [XX]:US
State or Province Name (full name) []:Arizona
Locality Name (eg, city) [Default City]:Phoenix
Organization Name (eg, company) [Default Company Ltd]:phoenixNAP
Organizational Unit Name (eg, section) []:PNAP
Common Name (eg, your name or your server's hostname) []:yourdomain.com
Email Address []:validemail@pnap.com

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

6. Proceed to configure the virtual host to display the new certificate. The process is identical to the steps outlined in Chapter 2, Configure Virtual Hosts for SSL.

7. Test your Apache configuration before restarting. To make sure that the syntax is correct, type:

```
sudo apachectl configtest
```

8. Once the system confirms that the syntax is correct, restart Apache:

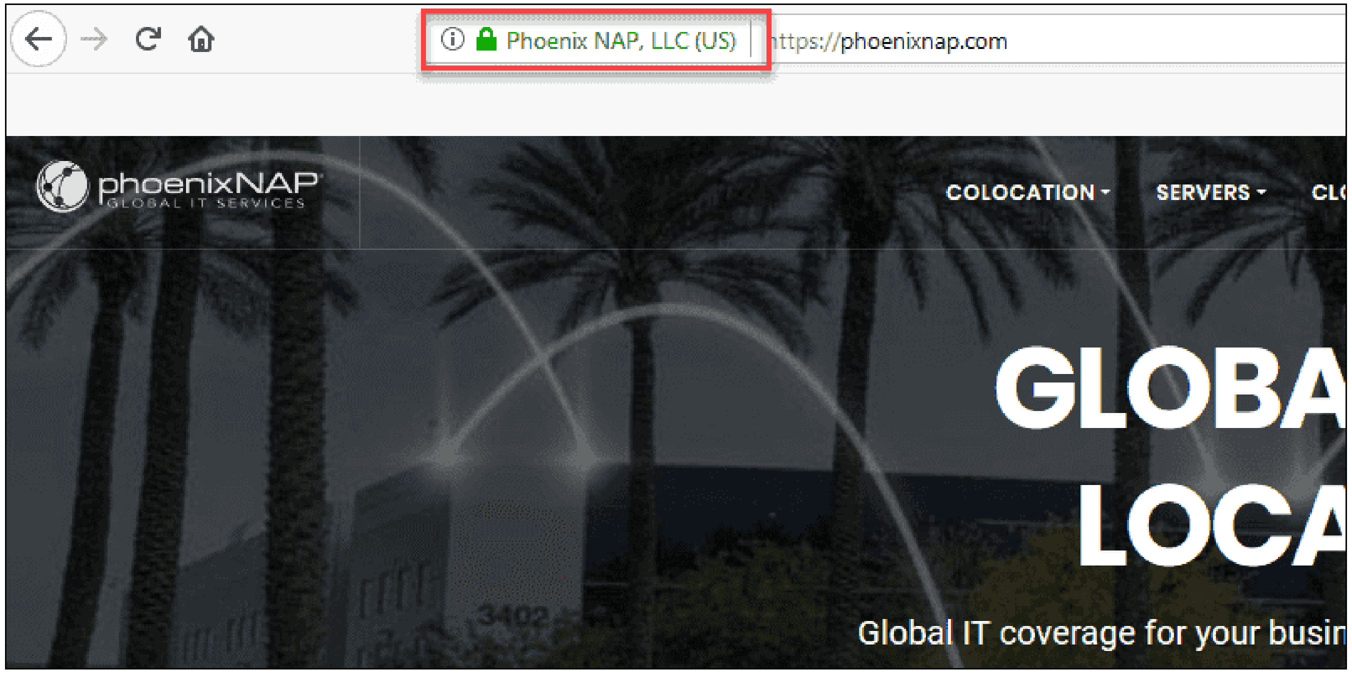
```
sudo systemctl restart httpd
```

You have now set up your Apache server to use your self-signed SSL certificate and should be able to visit your site with SSL enabled.

HOW TO CHECK IF A SSL CERTIFICATE IS VALID?

To check if a SSL Certificate is valid you can publically available services, such as the SSL Server Test. Confirm the status of your certificate, and to check if all the details are correct.

Alternatively, access your website using **https://** to see if the SSL certificate is visible. The green padlock indicates that the additional layer of encryption is present.



Conclusion

By following these instructions, you have secured traffic on your CentOS Linux distribution website by implementing an SSL Certificate.

Your new SSL certificate ensures that all data passing between the web server and browsers remain private and secure.