

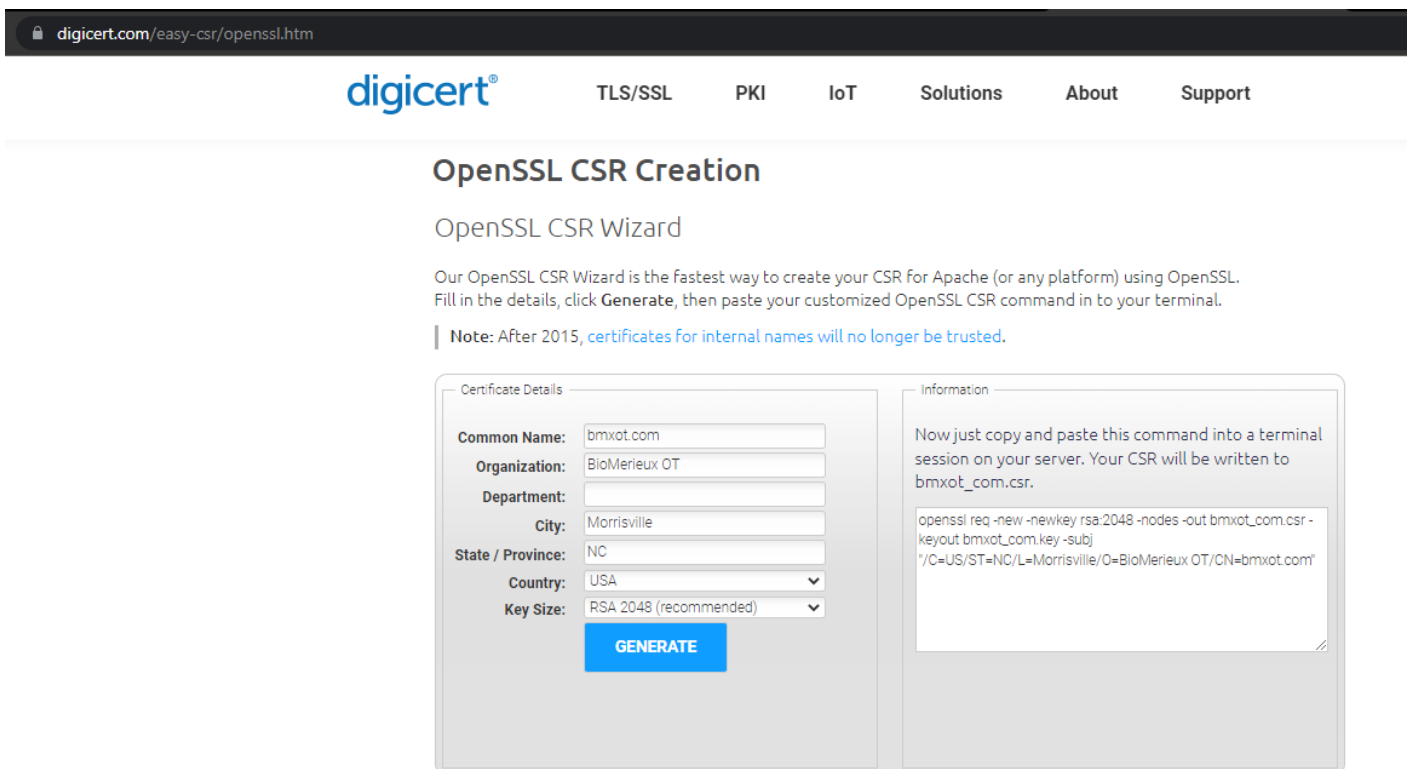
Generate SSL Cert

Here's some steps on how to generate and setup an SSL certificate for a web site.

1. First, order a certificate from a CA, like Sectigo, or DigiCert.
2. Once your order is active, they will require you to submit a CSR.
This must be generated by you, and will output two things.
It will generate your private key (for all SSL traffic). And, it will generate the CSR that contains your public key to be signed by the CA.
3. To generate a CSR is pretty easy. There's a wizard tool, here, that will give you the command line string that you can execute on a linux box using openssl.

[OpenSSL CSR Tool - Create Your CSR Faster | DigiCert.com](#)

The CSR wizard looks like this:



The screenshot shows the DigiCert website's OpenSSL CSR Creation wizard. The page title is "OpenSSL CSR Creation" and the sub-header is "OpenSSL CSR Wizard". Below the header, there is a note: "Our OpenSSL CSR Wizard is the fastest way to create your CSR for Apache (or any platform) using OpenSSL. Fill in the details, click **Generate**, then paste your customized OpenSSL CSR command in to your terminal." A note below that states: "Note: After 2015, certificates for internal names will no longer be trusted." The main form is divided into two sections: "Certificate Details" and "Information". The "Certificate Details" section contains the following fields: "Common Name" (bmxot.com), "Organization" (BioMerieux OT), "Department" (empty), "City" (Morrisville), "State / Province" (NC), "Country" (USA), and "Key Size" (RSA 2048 (recommended)). A blue "GENERATE" button is located below these fields. The "Information" section contains the text: "Now just copy and paste this command into a terminal session on your server. Your CSR will be written to bmxot_com.csr." Below this text is a code block containing the following command:

```
openssl req -new -newkey rsa:2048 -nodes -out bmxot_com.csr -keyout bmxot_com.key -subj "/C=US/ST=NC/L=Morrisville/O=BioMerieux OT/CN=bmxot.com"
```

Here's what the CSR wizard looks like for a wildcard domain:

NOTE: We've had some trouble with the Sectigo validation flow, in that the validation will report failure (on the web page), when the certificate actually got issued.

So, don't discard the private key you created, until you've confirmed via email that your CSR failed validation.

On a previous occasion, we saw the validation fail in the UI, and deleted the private key, to redo the CSR submission. But, it had actually passed and the certificate was issued... but the key was then deleted.

So, keep all versions until you have a validated key-crt pair.

6. Open the CSR file. It should look something like this:

```
GNU nano 6.2 b
-----BEGIN CERTIFICATE REQUEST-----
MIICoTCCAYkCAQAwxDELMAkGAlUEBhMCVVMxCzAJBgNVBAGMAk5DMRQwEgYDVQQH
DAcNb3JyaXN2aWxsZTEWMBQGA1UECgwNQmlvTWVyaWVleCBPVDESMBAGAlUEAwWJ
Yml4b3QuY29tMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAuizx2V01
dyueZYgCEfjixWGDJrUIGqGfFLwrHoDpgePz1ufgb5xQ+PUtI2DEU9gVJQvhjQf2
Hfd8zWP7L853rDhMAeoddkine+fmKLf1S71H+SPsyIK1DMVVF3fZMmmJtxhMgJXu
0bmSdvj/IULrn0LsGSjhxzPlaUDHkxkpwTg2CMHES2A0jA7MsFQheN1jWuojvDX5
ZcxOTlrcjMJ1GtaVV6M13ifBpMUE4C57xo/L30NLvIGB5CAgj5coFzSarU1AQfRL
W7YNIKe0KyrAz5+a26G8puN8QHxzzb6zJPhy5BYZ4bjS7WxTr6h2MCDgpr5wqRgL
7/3oNeNqu/IKnQIDAQABoAAwDQYJKoZIhvcNAQELBQADggEBAKPLdkaB2t9Jfnv7
tQHQredYWBDV4sE6ZbjxhNorv5ea46VXD1v9CcL02QxOrMbcNfKhJI3s9r9SiB4W
4PjA6MmvS190yCitcCV2cds+mj4a71U69Vvm/U8iwRoU/VmXVVIuStXiMQvcliOb
BZY0e7wi8GV/PBlzGL0b3WvM6e6Oqe8JPnKRmMBvJEH/QIHM/gkzvLHCX2eWAZL7
3Fz42gjWuV63cNkK9eJalbBSCV2t17nQjWk+rj26Lrr8bjKqfe6YCcrWJPdfZsBO
yqdyz/dOq0f8bbe/Zltt4WaJYSygrhBwmwcv sazufekxMfJw/U2NDFoQ35d52KGe
UN0ii5o=
-----END CERTIFICATE REQUEST-----
```

Paste its contents into a notepad session, so you can easily submit it to the CA website.

7. Go back to your CA website's setup flow, and submit your CSR.



Once submitted, your CA will require some form of proof (validation) that you own the domain they're creating the SSL key for. This will be via email reply, a custom DNS entry, or a file to expose on your website.

8. Go through the validation of that, and your CA will send you your signed SSL certificate and bundle.

For example: The validation flow with Sectigo has an option for DNS registration.

To use it, you must log into your domain registrar, and add a special CNAME record to your domain, that Sectigo will use to verify that you own the domain being registered.

A CNAME DNS entry for validating domain ownership with Sectigo looks like this:

<input type="checkbox"/>	CNAME	_ca5f45cf7b09f17bef0dd7ef08779ad3	2379eb40d97a8e838394632baba2d627.d4136b424bd59eae1 2cff64805da00da.sectigo.com.	1 Hour		
<input type="checkbox"/>	CNAME	_ca5f45cf7b09f17bef0dd7ef08779ad3.ogso fttech.com	2379eb40d97a8e838394632baba2d627.d4136b424bd59eae1 2cff64805da00da.sectigo.com.	1 Hour		

NOTE: It's not clear if the Name field (of the DNS record) should include the domain at the end of the hash string. So, we added two entries; one with, and one without.

- Once the validation steps have passed, the CA will issue your certificate (crt file) and a ca bundle.
- Download the crt and bundle files from your CA.
Now, you should have a key file (the private key you created with your CSR), a crt file (your public certificate), and your CA should have given you a CA bundle file as well.
- For Nginx to use your certificate, it must be chained with the CA bundle.
This is a simple concatenation operation. To so, make sure both crt and CA bundle files are in the same folder, and execute the following:
- To make one, run the following command:

```
$ cat www.sitecertificatefile.com.crt intermediatebundle.crt > www.example.com.chained.crt
```

The above command takes the site certificate file and adds on the intermediate certificate, putting both into a composite file, called a chained certificate.

NOTE: The order of concatenation is important in the above command, as NGINX will consider the first certificate in the chained file as your SSL certificate.

When NGINX starts up, it will attempt to match the key and first certificate in the chained file. If they don't match, it will give you an error like this:

```
SSL_CTX_use_PrivateKey_file("... /www.example.com.key") failed
(SSL: error:0B080074:x509 certificate routines:
X509_check_private_key:key values mismatch)
```

This chained certificate is what needs to be pushed into the NGINX server configuration.

Refer to this: [Configuring HTTPS servers](#)

- Move the key and chained crt files to your Nginx host, and configure Nginx to use them.

Updated 10 June 2025 05:12:11 by glwhite