



Generate a Self-Signed Certificate or a Certificate Signing Request

17 August 2023

IBM SevOne NPM Version 6.6.0

Document Version 6.6.0.0

Table of Contents

1 About	2
2 Product / Version	3
3 Common Scenarios	4
3.1 Generate a Private Key and a Self-Signed Certificate	4
3.2 Generate a Private Key and a Certificate Signing Request (CSR) for a Certificate Authority	5
3.3 Generate a CSR with a Subject Alternative Name (SAN)	5
4 Verify Certificate And Key	8
5 Move Files	9
6 Restart	10
7 Addendum - Intermediate Certificate	11
8 Other Useful Commands	12
9 Resources	13

SevOne Documentation

All documentation is available from the [IBM SevOne Support customer portal](#).

© Copyright International Business Machines Corporation 2023.

All right, title, and interest in and to the software and documentation are and shall remain the exclusive property of IBM and its respective licensors. No part of this document may be reproduced by any means nor modified, decompiled, disassembled, published or distributed, in whole or in part, or translated to any electronic medium or other means without the written consent of IBM.

IN NO EVENT SHALL IBM, ITS SUPPLIERS, NOR ITS LICENSORS BE LIABLE FOR ANY DAMAGES, WHETHER ARISING IN TORT, CONTRACT OR ANY OTHER LEGAL THEORY EVEN IF IBM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, AND IBM DISCLAIMS ALL WARRANTIES, CONDITIONS OR OTHER TERMS, EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, ON SOFTWARE AND DOCUMENTATION FURNISHED HEREUNDER INCLUDING WITHOUT LIMITATION THE WARRANTIES OF DESIGN, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT.


IBM, the IBM logo, and SevOne are trademarks or registered trademarks of International Business Machines Corporation, in the United States and/or other countries. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on ibm.com/trademark.

1 About

The creation and use of new SSL certificates is a part of establishing encrypted communications at most customer sites. This involves using signed certificates from an authority, operated or contracted by our customers. SSL certificates are used by the web server daemons on your server. They allow a Certificate Authority to *vouch* for your authenticity. This is designed to prevent an unauthorized user to intercept your browser's web page requests as they are sent from your computer to the web server, and pretends to be the webpage itself for malicious purposes such as stealing credit card numbers, identity, etc.

A **Certificate Authority (CA)** is someone that everyone on the internet or intranet trusts. It confirms the authenticity of the website and informs the user's browser of its authenticity. The process is pretty simple.

- You start by creating a **key** for your server. This is your server's personal identifier and is unique to your server. As an analogy, think of it as your server's Birth Certificate. Then, you can take this key to generate a **Certificate Signing Request (CSR)**. A CSR is brought to a CA as a request to apply for a **Digital Identity Certificate (CRT)**. A CRT is sort of like an ID card that says who you are.
- For servers connected to the internet (public-facing), this will typically be an internet company like Symantec, GoDaddy, Digi-cert, etc.
- For servers on an intranet (internal network), some larger companies might have their own CA to vouch for the legitimacy of their internal webpages to their internal customers.
- The CA takes the CSR which contains the unique key for your server and generates a CRT file for the requester.
- The CRT file is placed on the server and plugged into the web server configuration.
- Now, when a user requests a website from the server, the digital identity certificate is handed out to users as part of the request. The user's browser will use the CRT to ask the CA to confirm the server prior to loading the page. This is done automatically in the background by the browser. It also add the **lock** icon in your browser confirming that you have a secure and verified connection to the real server.

 There is an alternative to the CA process known as a **Self Signed Certificate**. Instead of a third party confirming your server's authenticity, your server validates itself.
Server: I'm me.
Browser: Says who? Server: Says me!"
This is good in that if your browser has been to the server and saved the self signed certificate in the past, and someone attempts to attack your browser, it will return that the certificate does not match.
With this method, you do not have a third-party confirming your server for the first time when the browser first saves the certificate, or subsequent times. This makes it more vulnerable.

2 Product/Version

Product	Version
SevOne NMS	6.x
	5.7.2.x

3 Common Scenarios

Commonly, there are two scenarios:

1. Generate a Private Key and a Self-Signed Certificate
2. Generate a Private Key and a Certificate Signing Request (CSR) for a Certificate Authority

You need to provide the web server(s) the **key** and **crt** in order to allow **https** webpages from your server. SevOne NMS 5.7.2 and up only use **nginx**.

- i** If the Apache keys are in /etc/ssl/apache2/, then **nginx** will be able to use them as well. When creating a certificate, there is an option to encrypt it with a password. The use of passwords on the certificates is discouraged because:
- the password will need to be typed in any time Apache is started or restarted
 - easy to forget the password and when the whole server is restarted, it will halt when loading the Apache service and it will not come back up completely until the correct password is entered at the prompt

If you have a customer with an encrypted CRT file, please see section **Other Useful Commands** below on how to remove the password if the customer agrees with that option.

3.1 Generate a Private Key and a Self-Signed Certificate

Generate key & certificate


```
$ openssl req -x509 -nodes -sha256 -days 365 -newkey rsa:2048 -keyout server.key -out server.crt
```

The output prompts for a Distinguished Name (DN) which is general information about the web server and the actual answers for a self-signed certificate do not matter too much.

i Example of output and some responses entered (see in bold)

```
Generating a 2048 bit RSA private key
.....+++
.....+++
writing new private key to 'server.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]:US
State or Province Name (full name) [Some-State]:Delaware
Locality Name (eg, city) []:Newark
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SevOne
Organizational Unit Name (eg, section) []:Support
Common Name (e.g. server FQDN or YOUR name) []:www.sevone.com
Email Address []:admin@sevone.com
```

3.2 Generate a Private Key and a Certificate Signing Request (CSR) for a Certificate Authority

 Avoid offering the customer the challenge password unless the customer absolutely needs it for their CA.

Generate CSR

```
$ openssl req -nodes -sha256 -days 365 -newkey rsa:2048 -keyout server.key -out server.csr
```

Example of output and some responses entered (see in bold)

```
Generating a 2048 bit RSA private key
.....+++
.....+++
writing new private key to 'server.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]:US
State or Province Name (full name) [Some-State]:Delaware
Locality Name (eg, city) []:Newark
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SevOne
Organizational Unit Name (eg, section) []:Support
Common Name (e.g. server FQDN or YOUR name) []:www.sevone.com
Email Address []:admin@sevone.com

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

3.3 Generate a CSR with a Subject Alternative Name (SAN)

An extension to the X.509 specification called **Subject Alternative Name (SAN)** allows for the definition of multiple host names on an SSL certificate. SANs are replacing common names on SSL certificates, and as of Google Chrome version 58, it is the only extension used to match the domain name and site certificate. A CSR can be generated with any number of SANs by following the method below.

1. Create a configuration file named **san.cnf**
2. Using a text editor of your choice, copy the following text into it, replacing the SevOne info in each line of the **req_distinguished_name** section with the appropriate information for your SSL certificate.

Example

```
[ req ]
```

```

default_bits = 2048
distinguished_name = req_distinguished_name
req_extensions = req_ext
default_days = 365
[ req_distinguished_name ]
countryName = US
stateOrProvinceName = Delaware
localityName = Newark
organizationName = SevOne
organizationalUnitName = Support
emailAddress = support@sevone.com
commonName = www.sevone.com
[ req_ext ]
subjectAltName = @alt_names
[alt_names]
DNS.1 = www.sevone.com
DNS.2 = support.sevone.com
DNS.3 = portal.sevone.com
    
```

- A **commonName** must still be specified, and can match a **subjectAltName**.
- In this example, three **alt_names** are specified, meaning that the SSL Certificate must be valid for all three hostnames.
- There can be any number of **alt_names**, just add/remove DNS.# lines as needed.

3. Run the **openssl** command with the **-config** flag, and specify the san.cnf file.

```
$ openssl req -out server.csr -newkey rsa:2048 -nodes -keyout server.key -config san.cnf
```

i You will be presented with input prompts for each item under **req_distinguished_name**, and you must re-enter the values.

! Do not accept default values as it will result in an error.

Example of output

```


$ openssl req -out server.csr -newkey rsa:2048 -nodes -keyout server.key -config
san.cnf
Generating a 2048 bit RSA private key
.....+++
.....+++
writing new private key to 'server.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.
-----
    
```


Generate a Self-Signed Certificate or a Certificate Signing Request

```
US []:US
Delaware []:Delaware
Newark []:Newark
SevOne []:SevOne
Support []:Support
support@sevone.com []:support@sevone.com
www.sevone.com []:www.sevone.com
```

The CSR and key generated by this method will include the SANs you have specified. They can be used to obtain, verify, and install an SSL through the normal methods described in this guide.

4 Verify Certificate and Key

 Verify that the certificate and server keys match **md5** sums.

Using the commands below, verify that the output matches.

Verify certificate commands

```
$ openssl rsa -noout -modulus -in server.key | openssl md5;  
$ openssl x509 -noout -modulus -in server.crt | openssl md5;
```

The output should look something like the following and must be identical for both commands.

```
(stdin)= 93b826c032ab9358efc88da0a744dc79
```

5 Move Files

Files must be moved to an appropriate location based on NMS version.

CentOS 7 / CentOS 8 Stream

Commands to backup existing files and install the newly created ones

```
$ mv /etc/nginx/ssl/nginx.crt /etc/nginx/ssl/nginx.crt.old
$ mv /etc/nginx/ssl/nginx.key /etc/nginx/ssl/nginx.key.old
$ mv server.crt /etc/nginx/ssl/nginx.crt
$ mv server.key /etc/nginx/ssl/nginx.key
$ chmod 400 /etc/nginx/ssl/nginx.crt /etc/nginx/ssl/nginx.key
```

6 Restart


Restart nginx for the new certificate to take effect.

```
$ supervisorctl restart nginx
```

7 Addendum - Intermediate Certificate

If the customer is using an intermediate certificate, you will need to include the following in the normal **server.crt**. It is important to maintain the order in the command - **server** hash must be followed by the **intermediate** hash.

```
$ cat server.crt intermediate.crt > server.crt.new
```

 Sometimes **cat** includes new line characters in the hash. Please remove these. You may then make a backup of the **server.crt** and then, replace **server.crt** with **server.crt.new**

8 Other Useful Commands

Generate a CSR from an existing private key

```
$ openssl req -key server.key -new -out server.csr
```

Generate a CSR from an existing certificate and private key

```
$ openssl x509 -in server.crt -signkey server.key -x509toreq -out server.csr
```

Generate a Self-Signed Certificate from an existing private key

```
$ openssl req -x509 -sha256 -days 365 -new -key server.key -out server.crt
```

Decrypt a private key

```
$ openssl rsa -in encrypted.key -out decrypted.key
```

Decrypt a CSR

```
$ openssl req -in server.csr -noout -text
```

9 Resources

- <http://www.thegeekstuff.com/2009/07/linux-apache-mod-ssl-generate-key-csr-crt-file/>
- https://en.wikipedia.org/wiki/Certificate_signing_request
- https://en.wikipedia.org/wiki/Public_key_certificate
- https://en.wikipedia.org/wiki/Certificate_authority
- https://en.wikipedia.org/wiki/Self-signed_certificate
- <https://www.digitalocean.com/community/tutorials/openssl-essentials-working-with-ssl-certificates-private-keys-and-csrs>
- <https://www.openssl.org/docs/man1.0.2/man1/>
- <https://support.google.com/chrome/a/answer/7391219?hl=en>
- <https://support.citrix.com/article/CTX227983>
- <https://support.dnsimple.com/articles/what-is-ssl-san/>