Toolkit Certificates from GoDaddy

This document is rich with examples taken from an actual MVSTK keystore generation with certificates from GoDaddy. For security, some of the included examples have been partially obfuscated. Several other substitutions need to be made as well:

yourDomain.com this is the domain reference for toolkit requests:

<https://yourDomain.com:8181/ws/wsr>

yourName for things like keystore file name et al.

yourPassword for things like the certificate and keystore password

**Note: these should ALWAYS be the same as each other**

yourDirectory the directory in which you will create everything

GoDaddy.pem the .pem file sent to you by GoDaddy

shown below as Blahblahblah.pem

Also, you’ll need to install the 32- or 64-bit versions of OpenSSL and the Java JDK v1.8. You’ll need 32-bit if you’re using Toolkit version 2.2.3 or lower and 64-bit if you’re using Toolkit version 2.3.1 or higher. Install them wherever you want. For the purposes of this document, the paths will be shown as follows:

C:\OpenSSL

C:\Java

GoDaddy will need two files: a .key file ( contains a private key ) and a .csr file ( certificate request )

.KEY File Generation:

openssl genrsa -des3 -out yourName.key 2048

password: yourPassword

Here’s what yourName.key looks like after you create it:

-----BEGIN RSA PRIVATE KEY-----

Proc-Type: 4,ENCRYPTED

DEK-Info: DES-blah-CBC,blahblahblah9A5EDC712

1vU9g9F3oFZ8jLzQWT+aarCxrCb4gurK3ViZbOYd//tXGYwqtNygwsHWWvK+NTMj

l5V2zH8CIhBPQo8m7gPyxWVVw8O+XHDOGMx1beqmxOWqQG6HtEiqFdvds+8KEnIg

( portion removed for security )

I0anJb8yCMzSIHShz1OTL2GIrbci/Cf2Q9Agz3XL0JyXEr406G6BrzfS+qzf31sK

+tA7+UQWw6XD7yUHzht1WoXxoPwru1g7TOHHNCAOkEtRyr/phCkR4xKsL0mMl3v4

V8HOFdoOMdh56UpDYCU4UemQPgbT99jziNa5jXJBieXSMz0ubbCaVVcgXs0TZvZD

-----END RSA PRIVATE KEY-----

.CSR File Generation:

openssl req -new -key yourName.key -out yourName.csr

password: yourPassword

Here’s what yourName.csr looks like after you create it:

-----BEGIN CERTIFICATE REQUEST-----

MIICszCCAZsCAQAwbjELMAkGA1UEBhMCRUMxEjAQBgNVBAgMCVBJQ0hJTkNIQTEO

MAwGA1UEBwwFUVVJVE8xDzANBgNVBAoMBkFsaW1lYzERMA8GA1UECwwIU2lzdGVt

YXMxFzAVBgNVBAMMDnd3dy5hbGltZWMuYXBwMIIBIjANBgkqhkiG9w0BAQEFAAOC

( portion removed for security )

IXRNzK1tGKuMDYbz9T4l97hEkgBhWNX8oJ9Fi5X0AhiDlejiVVJrJp+Yw7MMzaMO

IaP0vXmLCXoAZRxMQSBEUrNvyk3fWKAXE4KayJec24oAX6zWyQotBwzyqcns+Dg1

A7MCUWe4QEspZ7dlkv2FHaEYGaVrS/4=

-----END CERTIFICATE REQUEST-----

GoDaddy send you back:

Blahblahblah.crt Security certificate file

Blahblahblah.pem .PEM file ( this is the one you’ll use )

Gd\_bundle-g2-g1.crt Another Security certificate file

Creating the .PEM file for your keystore:

* Launch a Windows Command Prompt “as Administrator”
* CD \yourDirectory
* notepad yourName.pem
* open the yourName.key file with notepad
* open the GoDaddy.pem file with notepad
* copy the contents of yourName.key and paste into yourName.pem
* copy the contents of GoDaddy.pem and paste into yourName.pem below the key
* exit notepad making sure you save the file

This is what the yourName.pem file should look like:

-----BEGIN RSA PRIVATE KEY-----

Proc-Type: 4,ENCRYPTED

DEK-Info: DES-blah-CBC,blahblahblah9A5EDC712

1vU9g9F3oFZ8jLzQWT+aarCxrCb4gurK3ViZbOYd//tXGYwqtNygwsHWWvK+NTMj

l5V2zH8CIhBPQo8m7gPyxWVVw8O+XHDOGMx1beqmxOWqQG6HtEiqFdvds+8KEnIg

( portion removed for security )

I0anJb8yCMzSIHShz1OTL2GIrbci/Cf2Q9Agz3XL0JyXEr406G6BrzfS+qzf31sK

+tA7+UQWw6XD7yUHzht1WoXxoPwru1g7TOHHNCAOkEtRyr/phCkR4xKsL0mMl3v4

V8HOFdoOMdh56UpDYCU4UemQPgbT99jziNa5jXJBieXSMz0ubbCaVVcgXs0TZvZD

-----END RSA PRIVATE KEY-----

-----BEGIN CERTIFICATE-----

MIIGjTCCBXWgAwIBAgIIS4uZ/DpZQjcwDQYJKoZIhvcNAQELBQAwgbQxCzAJBgNV

BAYTAlVTMRAwDgYDVQQIEwdBcml6b25hMRMwEQYDVQQHEwpTY290dHNkYWxlMRow

( portion removed for security )

1mYbPTMLpmPSOfnKku9YleH4ZVzhtOi0naFNrD/HuMORGy6Zc7NhzCZxS9vuONQR

/i/mmHakL83N+pmaguyKBKhJWfBTKjS+DTPdLWcIxLIbWo+fSIf/oXmjucMOmO59

1Q==

-----END CERTIFICATE-----

Examine the Certificate:

C:\OpenSSL\bin>openssl x509 -in c:\yourDirectory\yourName.pem -noout -text

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

4b:blahblah:37

Signature Algorithm: sha256WithRSAEncryption

Issuer: C = US, ST = Arizona, L = Scottsdale, O = "GoDaddy.com, Inc.", OU = http://certs.godaddy.com/repository/, CN = Go Daddy Secure Certificate Authority - G2

Validity

Not Before: Aug 10 23:52:29 2021 GMT

Not After : Aug 21 13:32:19 2022 GMT

Subject: CN = www.yourDomain.com

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public-Key: (2048 bit)

Modulus:

00:bd:e7:af:5d:6e:7b:1f:27:53:03:c1:46:32:6c:

37:27:06:65:bd:fb:a1:76:56:36:56:ae:36:92:99:

( portion removed for security )

b1:c9:c4:a5:e2:71:7c:46:4c:70:f0:84:95:d7:43:

7d:88:cf:5e:45:14:15:aa:36:02:d8:df:d3:f1:3a:

83:7f

Exponent: 65537 (0x10001)

X509v3 extensions:

X509v3 Basic Constraints: critical

CA:FALSE

X509v3 Extended Key Usage:

TLS Web Server Authentication, TLS Web Client Authentication

X509v3 Key Usage: critical

Digital Signature, Key Encipherment

X509v3 CRL Distribution Points:

Full Name:

URI:http://crl.godaddy.com/blahblah.crl

X509v3 Certificate Policies:

Policy: 2.16.840.1.114413.1.7.23.1

CPS: http://certificates.godaddy.com/repository/

Policy: 2.23.140.1.2.1

Authority Information Access:

OCSP - URI:http://ocsp.godaddy.com/

CA Issuers - URI:http://certificates.godaddy.com/repository/gdig2.crt

X509v3 Authority Key Identifier:

keyid:40:C2:BD:27:8E:blahblah:FB:6C:B3:F0:B4:2C:80:CE

X509v3 Subject Alternative Name:

DNS:www.yourDomainmvs.com, DNS:yourDomainblah.com

X509v3 Subject Key Identifier:

69:59:65:46:21:blahblahblah7C:43:0E:EB:CF:B1

CT Precertificate SCTs:

Signed Certificate Timestamp:

Version : v1 (0x0)

Log ID : 29:79:BE:F0:9Eblahblah63:A5:77:E5:

BE:57:7D:9C:60blahblah26:5C:25:5D:C7:84

Timestamp : Aug 10 23:52:30.264 2021 GMT

Extensions: none

Signature : ecdsa-with-SHA256

30:44:02:20:37:93:67:CB:40:83:11:80:D0:49:11:34:

( portion removed for security )

DC:07:88:4A:D0:50:AF:41:76:90:F2:D9:D8:40:81:9D:

7A:0C:C7:74:44:34

Signed Certificate Timestamp:

Version : v1 (0x0)

Log ID : DF:A5:5E:blahblahEE:B8:5F:4E:3E:5A:

EA:CD:A2:12:A4:6A:5E:8E:3B:12:C0:20:44:5C:2A:73

Timestamp : Aug 10 23:52:30.815 2021 GMT

Extensions: none

Signature : ecdsa-with-SHA256

30:46:02:21:00:B4:28:4D:7A:4E:25:9E:26:41:73:32:

( portion removed for security )

EA:E4:B0:77:25:36:B9:CC:14:1E:5A:5F:10:B3:11:57:

B8:66:40:7C:04:76:CA:BD

Signed Certificate Timestamp:

Version : v1 (0x0)

Log ID : 41:C8:blahblah:10:C6:A1:3A:09:42:87:5E:

4E:31:8B:1B:03:EB:EB:4B:C7:68:F0:90:62:96:06:F6

Timestamp : Aug 10 23:52:30.965 2021 GMT

Extensions: none

Signature : ecdsa-with-SHA256

30:45:02:21:00:E2:EB:9F:B6:40:E9:0C:FD:E3:52:59:

( portion removed for security )

CD:28:64:2B:0D:5E:81:62:3C:9C:E0:E0:32:F1:0F:F9:

3D:16:22:26:15:DB:C9

Signature Algorithm: sha256WithRSAEncryption

18:4b:11:fd:3e:6d:ad:26:65:7a:03:38:2c:be:9e:86:5e:91:

af:45:d2:5e:26:e1:25:19:16:d8:c4:54:9b:5f:ed:2b:ad:7e:

( portion removed for security )

fa:99:9a:82:ec:8a:04:a8:49:59:f0:53:2a:34:be:0d:33:dd:

2d:67:08:c4:b2:1b:5a:8f:9f:48:87:ff:a1:79:a3:b9:c3:0e:

98:ee:7d:d5

Create .pkcs12 certificate:

C:\OpenSSL\bin>openssl pkcs12 -export -out c:\yourDirectory\yourName.pkcs12 -in c:\yourDirectory\yourName.pem

Enter pass phrase for c:\yourDirectory\yourName.pem

Enter Export Password: yourPassword

Verifying - Enter Export Password: yourPassword

Create an empty keystore:

Use the Administrative command prompt to create the C:\yourDirectory\yourName.bat file

SET ALIAS=RocketMVS

SET KEYSTORE\_PASSWORD=yourPassword

SET PRIVATE\_KEY\_PASSWORD=yourPassword

SET RESULT\_FILE=yourName.ks

SET COMMON\_NAME=RocketSoftwareOS

SET ORG\_UNIT=D3MV

SET ORG=Rocket Software

SET COUNTRY=US

SET DAYS\_BEFORE\_EXPIRE=360

IF EXIST %RESULT\_FILE% echo Deleting existing KeyStore file

IF EXIST %RESULT\_FILE% del %RESULT\_FILE%

@echo Generating KeyStore: %RESULT\_FILE%

"c:\Java\bin\keytool" -genkey -dname "cn=%COMMON\_NAME%, ou=%ORG\_UNIT%, o=%ORG%, c=%COUNTRY%" -alias %ALIAS% -keypass %PRIVATE\_KEY\_PASSWORD% -keystore %RESULT\_FILE% -storepass %KEYSTORE\_PASSWORD% -validity %DAYS\_BEFORE\_EXPIRE%

Then execute the .bat script to create the empty keystore file

C:\yourDirectory >yourName.bat

Output should look something like this:

C:\yourDirectory >SET ALIAS=RocketMVS

C:\yourDirectory >SET KEYSTORE\_PASSWORD=yourPassword

C:\yourDirectory >SET PRIVATE\_KEY\_PASSWORD=yourPassword

C:\yourDirectory >SET RESULT\_FILE= yourName.ks

C:\yourDirectory >SET COMMON\_NAME=RocketSoftwareOS

C:\yourDirectory >SET ORG\_UNIT=D3MV

C:\yourDirectory >SET ORG=Rocket Software

C:\yourDirectory >SET COUNTRY=US

C:\yourDirectory >SET DAYS\_BEFORE\_EXPIRE=360

C:\yourDirectory >IF EXIST yourName.ks echo Deleting existing KeyStore file

C:\yourDirectory >IF EXIST yourName.ks del yourName.ks

Generating KeyStore: yourName.ks

C:\yourDirectory>"c:\Java\bin\keytool" -genkey -dname "cn=RocketSoftwareOS, ou=D3MV, o=Rocket Software, c=US" -alias RocketMVS -keypass yourPassword -keystore yourName.ks -storepass yourPassword -validity 360

Warning:

The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard format using "keytool -importkeystore -srckeystore yourName.ks -destkeystore yourName.ks -deststoretype pkcs12".

Note: The warning is normal and means nothing to us.

Import the .pkcs12 certificate:

C:\Java\bin>keytool -v -importkeystore -srckeystore c:\yourDirectory\yourName.pkcs12 -srcstoretype PKCS12 -destkeystore c:\yourDirectory\yourName.ks -deststoretype JKS

Importing keystore c:\yourDirectory\yourName.pkcs12 to c:\yourDirectory\yourName.ks...

Enter destination keystore password: yourPassword

Enter source keystore password: yourPassword

Entry for alias 1 successfully imported.

Import command completed: 1 entries successfully imported, 0 entries failed or cancelled

[Storing c:\yourDirectory\yourName.ks]

Warning:

The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard format using "keytool -importkeystore -srckeystore c:\yourDirectory\yourName.ks -destkeystore c:\yourDirectory\yourName.ks -deststoretype pkcs12".

Note: The warning is normal and means nothing to us.

Install the keystore in the Toolkit server:

Place your newly-generated yourName.ks file in the appropriate directory on the MVS Toolkit server. The defaults are something like:

Windows: \Program Files\Rocket Software\mvs\server

Linux: /usr/local/RocketSoftware/mvs/server

Using the MVS Toolkit IDE Client:

* Connect to the Web Service Provider
* Double click the Web Service Provider to edit
* Click the “Use SSL Connection” checkbox
* Unclick the “Use Default Settings” checkbox
* Populate the following text boxes:
  + Key Store ( provide the path to yourName.ks on the server )
  + Key Store Password yourPassword
  + Confirm Key Store Password yourPassword
  + Key Password yourPassword
  + Confirm Key Password yourPassword
* Save the Web Service Provider
  + Click X then answer “Y” to “save” or
  + Click the floppy disk icon under “File”

Now restart the MVS Toolkit service

Windows: Service control manager > Rocket MV Web Service > stop

Service control manager > Rocket MV Web Service > start

Linux at root: service mv.serv stop

service mv.serv start