**MongoDB x509-client-authentication + Aspire**

**MongoDB using a Self-Signed Certificate:**

**References:**

**-** Attached File (Check the end of this file).

- [**https://docs.mongodb.com/manual/tutorial/configure-x509-client-authentication/**](https://docs.mongodb.com/manual/tutorial/configure-x509-client-authentication/)

- [**https://www.akadia.com/services/ssh\_test\_certificate.html**](https://www.akadia.com/services/ssh_test_certificate.html)

- [**https://stackoverflow.com/questions/991758/how-to-get-pem-file-from-key-and-crt-files**](https://stackoverflow.com/questions/991758/how-to-get-pem-file-from-key-and-crt-files)

- [**http://pe-kay.blogspot.com/2016/02/securing-mongodb-using-x509-certificate.html**](http://pe-kay.blogspot.com/2016/02/securing-mongodb-using-x509-certificate.html)

**Notes:**

- You can use **cygwin** to run **openssl** commands

- %PASSPHRASE% = 123456abC

**Generating CA**

1. $ openssl genrsa -des3 -passout pass:%PASSPHRASE% -out %LOCATION%/ca/ca.key 1024
2. $ openssl req -new -key %LOCATION%/ca/ca.key -passin pass:%PASSPHRASE% -out %LOCATION%/ca/ca.csr -subj "/C=US/ST=VA/L=Herndon/O=SearchTechnologies/**CN=TestCA**"
3. $ openssl x509 -req -days 3650 -in %LOCATION%/ca/ca.csr -out %LOCATION%/ca/ca.crt -signkey %LOCATION%/ca/ca.key -passin pass:%PASSPHRASE%
4. $ cat %LOCATION%/ca/ca.crt %LOCATION%/ca/ca.key > %LOCATION%/ca/ca.pem

**Generating Server Certificate**

1. $ openssl genrsa -des3 -passout pass:%PASSPHRASE% -out %LOCATION%/server/mongo.key 1024
2. $ openssl req -new -key %LOCATION%/server/mongo.key -passin pass:%PASSPHRASE% -out %LOCATION%/server/mongo.csr -subj "/C=US/ST=VA/L=Herndon/O=SearchTechnologies/**CN=localhost**"
3. $ openssl x509 -req -in %LOCATION%/server/mongo.csr -out %LOCATION%/server/mongo.crt -CA %LOCATION%\ca\ca.crt -CAkey %LOCATION%/ca/ca.key -passin pass:%PASSPHRASE% -CAcreateserial -days 3650
4. $ cat %LOCATION%/server/mongo.crt %LOCATION%/server/mongo.key > %LOCATION%/server/mongo.pem

**Generating Client Certificate**

1. $ openssl genrsa -des3 -passout pass:%PASSPHRASE% -out %LOCATION%/client/aspire.key 1024
2. $ openssl req -new -key %LOCATION%/client/aspire.key -passin pass:%PASSPHRASE% -out %LOCATION%/client/aspire.csr -subj "/C=US/ST=VA/L=Herndon/**O=SearchTechnologiesAspire**/**CN=aspire**"
3. openssl x509 -req -in %LOCATION%/client/aspire.csr -out %LOCATION%/client/aspire.crt -CA %LOCATION%/ca/ca.crt -CAkey %LOCATION%/ca/ca.key -passin pass:%PASSPHRASE% -CAcreateserial -days 3650
4. $ cat %LOCATION%/client/aspire.crt %LOCATION%/client/aspire.key > %LOCATION%/client/aspire.pem

**Starting up MongoDB**

>> mongod --sslMode requireSSL --sslPEMKeyFile <path to SSL certificate and key PEM file> --sslCAFile <path to root CA PEM file>

**ie:**

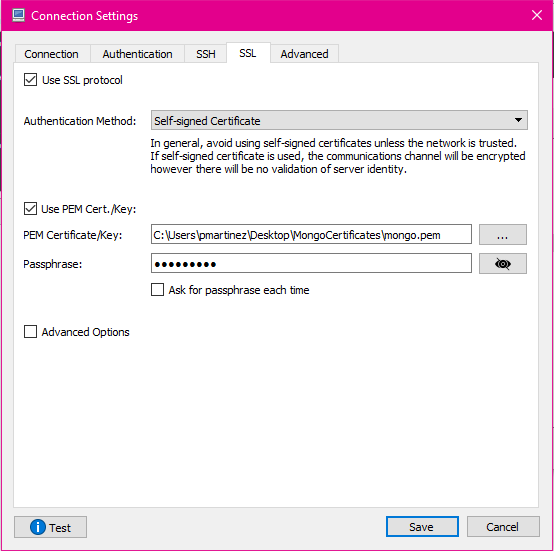
**>> mongod --sslMode requireSSL --sslPEMKeyFile %LOCATION%/server/mongo.pem --sslCAFile %LOCATION%/ca/ca.pem**

In order to start MongoDB on Windows, specify the sslPEMKeyPassword argument instead of wait to be prompted. For some reason it’s not working when you provide the password in the prompt.

mongod.exe --dbpath "c:\dev\mongo\3.4\data" --sslMode requireSSL --sslPEMKeyFile c:\dev\mongox509\server\mongo.pem --sslPEMKeyPassword %PASSPHRASE% --sslCAFile c:\dev\mongox509\ca\ca.pem

**Creating client user in MongoDB**

1. Open RoboMongo and connect using SSL connection.



1. Open any collection or a new console
2. USER should be the SUBJECT of the CLIENT certificate:

>> openssl x509 -in  %LOCATION%/client/aspire.pem -inform PEM -subject -nameopt RFC2253 | grep subject

C:\Users\PMARTI~1\AppData\Local\Temp\Image.png

1. Execute the following query:

db.getSiblingDB("$external").runCommand(

  {

    createUser: "CN=aspire,O=SearchTechnologiesAspire,L=Herndon,ST=VA,C=US",

    roles: [

             { role: 'userAdminAnyDatabase', db: 'admin' }

           ],

    writeConcern: { w: "majority" , wtimeout: 5000 }

  }

)

1. Check the query is successfully executed:



**Aspire Connecting to Mongo X.509 Authentication**

**References**:

[**https://search-tech.atlassian.net/wiki/display/aspire31/MongoDB+Settings**](https://search-tech.atlassian.net/wiki/display/aspire31/MongoDB+Settings)

1. Add the following in the settings.xml file

<noSQLConnectionProvider **sslEnabled="true"** sslInvalidHostNameAllowed="false">

    <implementation>com.searchtechnologies.aspire:aspire-mongodb-provider</implementation>

    <servers>localhost:27017</servers>

**<x509username>CN=aspire,O=SearchTechnologiesAspire,L=Herndon,ST=VA,C=US</x509username>**

</noSQLConnectionProvider>

**Note:**  If you don't know what to use into the <x509username> field execute the following command using the x509 client certificate:

>> openssl x509 -in  %LOCATION%/client/aspire.pem -inform PEM -subject -nameopt RFC2253 | grep subject

1. For using x509 authentication you need to import the client x509 certificate into a java keystore for Aspire to be able to present it to the server for authentication. (The truststore should already be set in the startup script for self-signed certificates)

For importing the x509 certificate (client.pem) into a java keystore you need to execute the following commands:

$ openssl pkcs12 -export -out %LOCATION%/client/aspire.pkcs12 -in %LOCATION%/client/aspire.pem

Enter Export Password: <your-password-here>

$ keytool-importkeystore -srckeystore %LOCATION%/client/aspire.pkcs12-srcstoretype PKCS12 -destkeystore%LOCATION%/client/aspire.jks -deststoretype JKS

Enter destination keystore password:

Re-enter new password: <your-password-here>

Enter source keystore password: <your-password-here>

Entry for alias 1 successfully imported.

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

3. After importing the client's certificate into a java keystore, you need to include it into the Aspire startup script (aspire.bat):

-Djavax.net.ssl.keyStore=%LOCATION%/client/aspire.jks

-Djavax.net.ssl.keyStorePassword=password

**Note:**  Since you have a Self-Signed Certificate and not a Java Trusted CA you will need to add also:

-Djavax.net.ssl.trustStore=%LOCATION%/ca/ca.jks

-Djavax.net.ssl.trustStorePassword=password

C:\Users\PMARTI~1\AppData\Local\Temp\Image.png

C:\Users\PMARTI~1\AppData\Local\Temp\Image.png

**Attached File**

|  |
| --- |
| :GotPassphrase  echo Generating CA  openssl genrsa -des3 -passout pass:%PASSPHRASE% -out %LOCATION%\ca\ca.key 1024 >nul 2>nul  openssl req -new -key %LOCATION%\ca\ca.key -passin pass:%PASSPHRASE% -out %LOCATION%\ca\ca.csr -subj "/C=US/ST=VA/L=Herndon/O=SearchTechnologies/CN=TestCA" -config openssl.cnf >nul 2>nul  openssl x509 -req -days 3650 -in %LOCATION%\ca\ca.csr -out %LOCATION%\ca\ca.crt -signkey %LOCATION%\ca\ca.key -passin pass:%PASSPHRASE% >nul 2>nul  echo Generating Server Certificate  openssl genrsa -des3 -passout pass:%PASSPHRASE% -out %LOCATION%\server\server.key 1024 >nul 2>nul  openssl req -new -key %LOCATION%\server\server.key -passin pass:%PASSPHRASE% -out %LOCATION%\server\server.csr -subj "/C=US/ST=VA/L=Herndon/O=SearchTechnologies/CN=localhost" -config openssl.cnf >nul 2>nul  openssl x509 -req -in %LOCATION%\server\server.csr -out %LOCATION%\server\server.crt -CA %LOCATION%\ca\ca.crt -CAkey %LOCATION%\ca\ca.key -passin pass:%PASSPHRASE% -CAcreateserial -days 3650 >nul 2>nul  echo Generating Client Certificate  openssl genrsa -des3 -passout pass:%PASSPHRASE% -out %LOCATION%\client\client.key 1024 >nul 2>nul  openssl req -new -key %LOCATION%\client\client.key -passin pass:%PASSPHRASE% -out %LOCATION%\client\client.csr -subj "/C=US/ST=VA/L=Herndon/O=SearchTechnologies/CN=%COMMONNAME%" -config openssl.cnf >nul 2>nul  openssl x509 -req -in %LOCATION%\client\client.csr -out %LOCATION%\client\client.crt -CA %LOCATION%\ca\ca.crt -CAkey %LOCATION%\ca\ca.key -passin pass:%PASSPHRASE% -CAcreateserial -days 3650 >nul 2>nul  openssl pkcs12 -export -clcerts -in %LOCATION%\client\client.crt -inkey %LOCATION%\client\client.key -passin pass:%PASSPHRASE% -passout pass:%PASSPHRASE% -out %LOCATION%\client\client.p12 >nul 2>nul |