TLS certificate

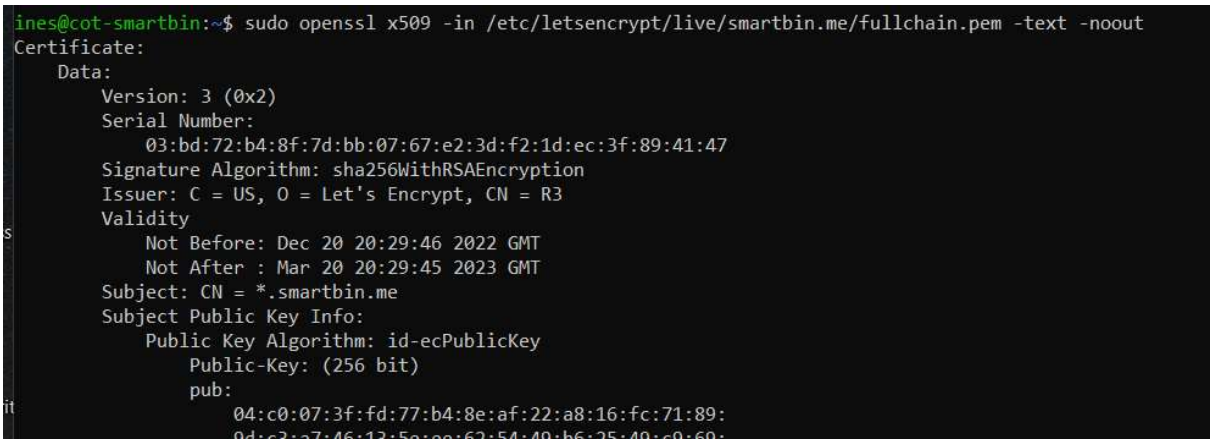
sudo certbot certonly --manual --agree-tos --no-eff-email --staple-ocsp --key-type=ecdsa --elliptic-curve=secp256r1 --preferred-challenges dns --debug-challenges -d *.smartbin.me -d smartbin.me
This command will output something similar to:

Please deploy a DNS TXT record under the name:

_acme-challenge.smartbin.me.

with the following value:

ypyQ92XlljLKZRMQONkqAKD1lKcFg9gIctc8PRkf2S8



```
            X509v3 Subject Alternative Name:
                DNS:*.smartbin.me, DNS:smartbin.me
            X509v3 Certificate Policies:
                Policy: 2.23.140.1.2.1
                Policy: 1.3.6.1.4.1.44947.1.1.1
                  CPS: http://cps.letsencrypt.org

            CT Precertificate SCTs:
                Signed Certificate Timestamp:
                    Version   : v1 (0x0)
                    Log ID    : 7A:32:8C:54:D8:B7:2D:B6:20:EA:38:E0:52:1E:E9:84:
                                16:70:32:13:85:4D:3B:D2:2B:C1:3A:57:A3:52:EB:52
                    Timestamp : Dec 20 21:29:46.753 2022 GMT
                    Extensions: none
                    Signature : ecdsa-with-SHA256
                                30:45:02:20:0E:18:DE:B8:C7:05:31:A2:9F:60:35:F5:
                                6B:DA:34:EB:2C:5E:2A:60:59:54:17:1D:C7:B8:67:AF:
```

We now have a valid certificate to use with all subdomains.

```
ines@cot-smartbin:~$ sudo openssl pkcs12 -export -out cert.pfx -inkey /etc/letsencrypt/live/smartbin.me/privkey.pem -in
/etc/letsencrypt/live/smartbin.me/cert.pem -certfile /etc/letsencrypt/live/smartbin.me/fullchain.pem
Enter Export Password:
Verifying - Enter Export Password:
```

```
ines@cot-smartbin:~$ sudo keytool -importkeystore -srckeystore cert.pfx -srcstoretype PKCS12 -srcstorepass azerty2wxcQSD
fgh -storepass azerty2wxcQSDfgh -destkeystore cert.jks -deststorepass azerty2wxcQSDfgh
Importing keystore cert.pfx to cert.jks...
Entry for alias 1 successfully imported.
Import command completed:  1 entries successfully imported, 0 entries failed or cancelled
```

MQTT Broker

Install Mosquitto: type the following commands, through install. mosquitto and mosquitto-clients will be installed:

$ sudo apt-get update

$ sudo add-apt-repository ppa:mosquitto-dev/mosquitto-ppa

$ sudo apt update

$ sudo apt-get install mosquitto

$ sudo apt-get install mosquitto-clients

Start Mosquitto:

$ sudo systemctl start mosquitto

$ sudo systemctl enable mosquitto

```
ines@cot-smartbin:~$ sudo systemctl enable mosquitto
Synchronizing state of mosquitto.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable mosquitto
```

```
ines@cot-smartbin:~$ sudo systemctl status mosquitto
● mosquitto.service - Mosquitto MQTT v3.1/v3.1.1 Broker
     Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-12-27 14:04:11 UTC; 8min ago
       Docs: man:mosquitto.conf(5)
             man:mosquitto(8)
   Main PID: 4543 (mosquitto)
      Tasks: 3 (limit: 9530)
     Memory: 1.4M
     CGroup: /system.slice/mosquitto.service
             └─4543 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

Dec 27 14:04:11 cot-smartbin systemd[1]: Starting Mosquitto MQTT v3.1/v3.1.1 Broker...
Dec 27 14:04:11 cot-smartbin mosquitto[4543]: 1672149851: Loading config file /etc/mosquitto/conf.d/default.conf
Dec 27 14:04:11 cot-smartbin mosquitto[4543]: [ 5109.534083]~DLT~ 4543~INFO     ~FIFO /tmp/dlt cannot be opened. Retryi
Dec 27 14:04:11 cot-smartbin systemd[1]: Started Mosquitto MQTT v3.1/v3.1.1 Broker.
```

```
ines@cot-smartbin:~$ mosquitto -version
Error: Unknown option '-version'.
mosquitto version 1.6.9

mosquitto is an MQTT v3.1.1 broker.
```

Configuring MQTT Password:

$ sudo mosquitto_passwd -c /etc/mosquitto/passwd middleware

```
ines@cot-smartbin:~$  sudo mosquitto_passwd -c /etc/mosquitto/passwd middleware
Password:
Reenter password:
```

Configuring MQTT settings:

$ sudo nano /etc/mosquitto/conf.d/default.conf

```
listener 1883
allow_anonymous false
password_file /etc/mosquitto/passwd

listener 8883
certfile /etc/mosquitto/certs/server.pem
cafile  /etc/mosquitto/certs/server.pem
keyfile /etc/mosquitto/certs/server.key

listener 8083
protocol websockets
certfile /etc/mosquitto/certs/server.pem
cafile /etc/mosquitto/certs/server.pem
keyfile /etc/mosquitto/certs/server.key
```

Restart the broker:

$ sudo systemctl restart mosquitto

```
ines@cot-smartbin:~$ sudo systemctl restart mosquitto
ines@cot-smartbin:~$ sudo systemctl status mosquitto
● mosquitto.service - Mosquitto MQTT v3.1/v3.1.1 Broker
     Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-12-27 14:34:25 UTC; 34s ago
       Docs: man:mosquitto.conf(5)
             man:mosquitto(8)
   Main PID: 5971 (mosquitto)
      Tasks: 3 (limit: 9530)
     Memory: 1.2M
     CGroup: /system.slice/mosquitto.service
             └─5971 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

Dec 27 14:34:25 cot-smartbin systemd[1]: Starting Mosquitto MQTT v3.1/v3.1.1 Broker...
Dec 27 14:34:25 cot-smartbin mosquitto[5971]: 1672151665: Loading config file /etc/mosquitto/conf.d/default.conf
Dec 27 14:34:25 cot-smartbin mosquitto[5971]: [ 6923.588897]~DLT~ 5971~INFO     ~FIFO /tmp/dlt cannot be opened. Retryi
Dec 27 14:34:25 cot-smartbin systemd[1]: Started Mosquitto MQTT v3.1/v3.1.1 Broker.
lines 1-15/15 (END)
```

Testing the changes:

mosquitto_sub -t “test” -u “<username>” -P “<Password>”

mosquitto_pub -t “test” -m “hello world” -u “<username>” -P “<Password>”

Link the VM to a subdomain:

Add the VM public IP address to the DNS records at Namecheap as an A record with the host as mqtt

configure SSL/HTTPS on WildFly

configure a key-store in the Elytron subsystem that will be used to hold your Let's Encrypt account key.

```
[standalone@localhost:9990 /] /subsystem=elytron/key-store=inbg:add(path=cert.jks,relative-to=jboss.server.config
.dir,credential-reference={clear-text="azerty2wxcQSDfgh"},type=JKS)
{"outcome" => "success"}
```

Server Keystore credentials

A key manager definition for creating the key manager list as used to create an SSL context

```
[standalone@localhost:9990 /] /subsystem=elytron/key-manager=inbgksm:add(key-store=inbg,credential-reference={cle
ar-text="HS8t2zclthRy3nSUbw"})
{"outcome" => "success"}
```

Server keystore Protocols

An SSL context for use on the server side of a connection.

```
[standalone@localhost:9990 /] /subsystem=elytron/server-ssl-context=inbgsslcontext:add(key-manager=inbgksm,protoc
ols=["TLSv1.3"],cipher-suite-names="TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256")
{"outcome" => "success"}
```

Store SSL Context information in undertow

```
[standalone@localhost:9990 /] /subsystem=undertow/server=default-server/https-listener=https:write-attribute(name
=ssl-context,value=inbgsslcontext)
{
    "outcome" => "success",
    "response-headers" => {
        "operation-requires-reload" => true,
        "process-state" => "reload-required"
    }
}
```

```
[standalone@localhost:9990 /] /core-service=management/management-interface=http-interface:write-attribute(name=s
sl-context,value=inbgsslcontext)
{
    "outcome" => "success",
    "response-headers" => {
        "operation-requires-reload" => true,
        "process-state" => "reload-required"
    }
}
```