一、mosquitto汤改版 学习（by cdq）

- 实验机：ubuntu20.04
- vmware
- 保存快照6.28

1. 卸载原来安装的mosquitto

```
//加密机卸载进入mosquitto安装文件 （make uninstall）然后卸载
//卸载mosquitto
sudo apt-get remove mosquitto
// 卸载mosquitto及其依赖文件
sudo apt-get remove --auto-remove mosquitto
//删除配置文件
sudo apt-get purge mosquitto
sudo apt-get purge --auto-remove mosquitt
//删除已安装的文件(whereis mosquitto)
sudo rm  /usr/local/sbin/mosquitto
sudo rm  -rf /etc/mosquitto
//删除可执行文件(whereis mosquitto_sub)
sudo rm /usr/local/bin/mosquitto_sub /usr/local/bin/mosquitto_pub
/usr/local/bin/mosquitto_passwd
sudo rm /usr/bin/mosquitto
```

2. 汤改版mosquitto的配置安装

2.1 安装TASSL

- 解压TASSL

  TASSL-1.1.1k-master.zip

- 配置

  > ./config --prefix=/opt/local no-shared
  >
  > 由于许多系统有自带的 ssl 库，为避免潜在的动态库冲突，此处仅生成静态库

- 编译安装

  > make
  >
  > sudo make install
  >
  > //使用sudo是因为会在/opt文件夹创建文件

- 检查TASSL版本

  > /opt/local/bin/openssl version -a

```
cdq@cdq-ubuntu:/opt$ /opt/local/bin/openssl version -a
OpenSSL 1.1.1k  Tassl 2.0.4 01 Dec 2021
built on: Tue Jun 28 03:14:31 2022 UTC
platform: linux-x86_64
options:  bn(64,64) rc4(16x,int) des(int) idea(int) blowfish(ptr)
compiler: gcc -fPIC -pthread -m64 -Wa,--noexecstack -Wall -O3 -DOPENSSL_USE_NODE
LETE -DL_ENDIAN -DOPENSSL_PIC -DOPENSSL_CPUID_OBJ -DOPENSSL_IA32_SSE2 -DOPENSSL_
BN_ASM_MONT -DOPENSSL_BN_ASM_MONT5 -DOPENSSL_BN_ASM_GF2m -DSHA1_ASM -DSHA256_ASM
 -DSHA512_ASM -DKECCAK1600_ASM -DRC4_ASM -DMD5_ASM -DAESNI_ASM -DVPAES_ASM -DGHA
SH_ASM -DECP_NISTZ256_ASM -DX25519_ASM -DPOLY1305_ASM -DNDEBUG
OPENSSLDIR: "/opt/local/ssl"
ENGINESDIR: "/opt/local/lib/engines-1.1"
Seeding source: os-specific
```

2.2 修改mosquitto汤改版

修改2.1 编译安装的TASSL的路径，主要有一下两处

- 修改config.mk 文件 line147

```
ifeq ($(UNAME),SunOS)
    ifeq ($(CC),cc)
        CFLAGS?=-O -I/root/tassl/include/openssl
    else
        CFLAGS?=-Wall -ggdb -O2 -I/root/tassl/include/openssl
    endif
else
    CFLAGS?=-Wall -ggdb -O2 -Wconversion -Wextra -
I/root/tassl/include/openssl
endif
LDFLAGS:= -pthread /root/tassl/lib/libssl.a /root/tassl/lib/libcrypto.a -lrt
STATIC_LIB_DEPS:=
```

```
ifeq ($(UNAME),SunOS)
    ifeq ($(CC),cc)
        CFLAGS?=-O -I/opt/local/include/openssl
    else
        CFLAGS?=-Wall -ggdb -O2 -I/opt/local/include/openssl
    endif
else
    CFLAGS?=-Wall -ggdb -O2 -Wconversion -Wextra -
I/opt/local/include/openssl
endif
LDFLAGS:= -pthread /opt/local/lib/libssl.a /opt/local/lib/libcrypto.a -lrt
STATIC_LIB_DEPS:=
```

- 修改config.mk文件 line 239-247

```
ifeq ($(WITH_TLS),yes)
    APP_CPPFLAGS:=$(APP_CPPFLAGS) -DWITH_TLS
    BROKER_CPPFLAGS:=$(BROKER_CPPFLAGS) -DWITH_TLS
    BROKER_LDADD:=$(BROKER_LDADD) /root/tassl/lib/libssl.a
/root/tassl/lib/libcrypto.a -ldl
    CLIENT_CPPFLAGS:=$(CLIENT_CPPFLAGS) -DWITH_TLS -ldl
    LIB_CPPFLAGS:=$(LIB_CPPFLAGS) -DWITH_TLS -ldl
    LIB_LIBADD:=$(LIB_LIBADD) /root/tassl/lib/libssl.a
/root/tassl/lib/libcrypto.a
    PASSWD_LDADD:=$(PASSWD_LDADD) /root/tassl/lib/libssl.a
/root/tassl/lib/libcrypto.a -ldl
    STATIC_LIB_DEPS:=$(STATIC_LIB_DEPS) /root/tassl/lib/libssl.a
/root/tassl/lib/libcrypto.a
```

将tassl相关路径进行修改

```
ifeq ($(WITH_TLS),yes)
    APP_CPPFLAGS:=$(APP_CPPFLAGS) -DWITH_TLS
    BROKER_CPPFLAGS:=$(BROKER_CPPFLAGS) -DWITH_TLS
    BROKER_LDADD:=$(BROKER_LDADD) /opt/local/lib/libssl.a
/opt/local/lib/libcrypto.a -ldl
    CLIENT_CPPFLAGS:=$(CLIENT_CPPFLAGS) -DWITH_TLS -ldl
    LIB_CPPFLAGS:=$(LIB_CPPFLAGS) -DWITH_TLS -ldl
    LIB_LIBADD:=$(LIB_LIBADD) /opt/local/lib/libssl.a
/opt/local/lib/libcrypto.a
    PASSWD_LDADD:=$(PASSWD_LDADD) /opt/local/lib/libssl.a
/opt/local/lib/libcrypto.a -ldl
    STATIC_LIB_DEPS:=$(STATIC_LIB_DEPS) /opt/local/lib/libssl.a
/opt/local/lib/libcrypto.a
```

2.3 安装mosquitto汤改版

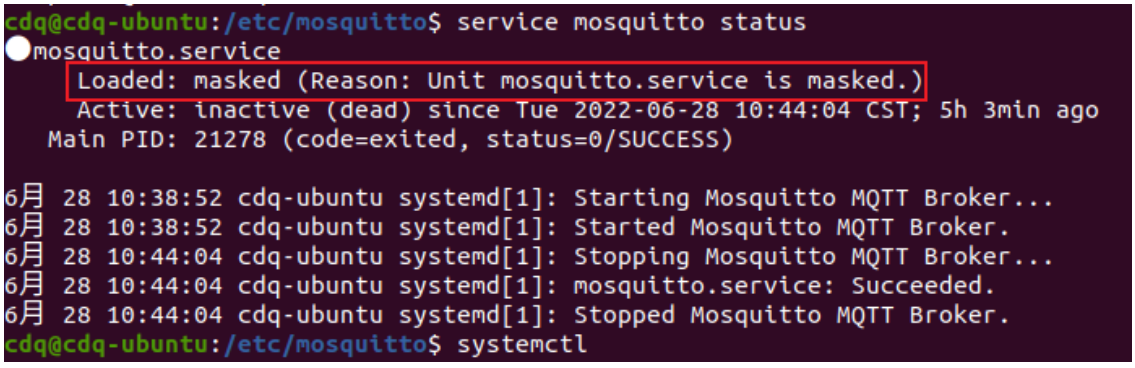
> make
>
> sudo make install

2.4 mosquitto 测试命令

- 查看mosquitto状态

  > service mosquitto status

  这里可能会遇到问题，显示Unit mosquitto.service is masked



  解决方案：调用命令`systemctl unmask mosquitto.service`，之后再开启mosquitto即可

  `service mosquitto start`

```
cdq@cdq-ubuntu:/etc/mosquitto$ systemctl unmask mosquitto.service
Removed /etc/systemd/system/mosquitto.service.
cdq@cdq-ubuntu:/etc/mosquitto$ service mosquitto start
cdq@cdq-ubuntu:/etc/mosquitto$ service mosquitto status
● mosquitto.service - LSB: mosquitto MQTT v3.1 message broker
     Loaded: loaded (/etc/init.d/mosquitto; generated)
     Active: active (exited) since Tue 2022-06-28 15:48:21 CST; 3s ago
       Docs: man:systemd-sysv-generator(8)
    Process: 64468 ExecStart=/etc/init.d/mosquitto start (code=exited, status=0>

6月 28 15:48:21 cdq-ubuntu systemd[1]: Starting LSB: mosquitto MQTT v3.1 messag>
6月 28 15:48:21 cdq-ubuntu systemd[1]: Started LSB: mosquitto MQTT v3.1 message>
lines 1-8/8 (END)
[2]+  已停止                  service mosquitto status
cdq@cdq-ubuntu:/etc/mosquitto$
```

如果没有systemctl指令，解决办法：

> 安装 systemctl
> sudo apt-get install --reinstall systemd
> 设置服务开机启动
> sudo systemctl enable ssh

- 查看mosquitto的进程

> ps -aux | grep mosquitto

- 杀掉进程

> kill -9 18248

- 启动mosquitto

> mosquitto -c /etc/mosquitto/mosquitto.conf -d

2.5 mosquitto 默认配置修改

主要是针对文件`/etc/mosquitto/mosquitto.conf`文件进行修改

1. 设置监听窗口`listener 1883`

```
214# listener port-number [ip address/host name/unix socket path]
215listener 1883
```

2. 关闭匿名用户登录 line512

```
509# Defaults to false, unless there are no listeners defined in the
configuration
510# file, in which case it is set to true, but connections are only allowed
from
511# the local machine.
512# allow_anonymous false
```

将512行，解除注释，设置为false

3. 设置用户名密码文件

```
527# See the TLS client require_certificate and use_identity_as_username options
528# for alternative authentication options. If an auth_plugin is used as well
as
529# password_file, the auth_plugin check will be made first.
password_file /etc/mosquitto/pwfile.example
```

将530行，解除注释，添加用户名密码文件的路径

4. 创建自定义用户

```
mosquitto_passwd -c /etc/mosquitto/pwfile.example user1
```

注意：这里使用 -c 会清除之前的所有用户，不适用 -c 表示添加用户。

用户信息存储于 `/etc/mosquitto/pwfile.example`，对密码进行了加密

5. 重启mosquitto服务生效

```
ps -aux | grep mosquitto
kill -9 [PID]
mosquitto -c /etc/mosquitto/mosquitto.conf -d
```

3. 汤改版mosquitto测试

1. 简单测试

- 一台客户端订阅主题topic：`mosquitto_sub -t topic`
- 一台客户端发布主题topic：`mosquitto_pub -t topic -m "hello"`
- 成功接收到发布的消息

2. 代码测试

▶ pub发布客户端pub.c
▶ sub订阅客户端sub.c

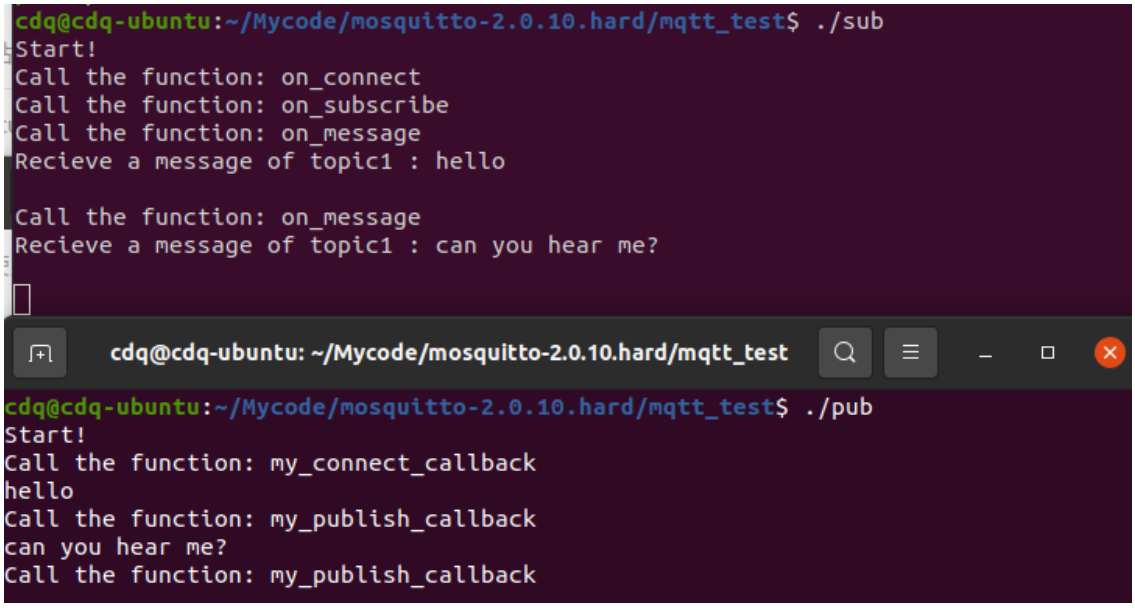
3. 运行测试

注意 这里使用汤改版的mosquitto，直接使用`gcc sub.c -o sub -lmosquitto`会报错

```
cdq@cdq-ubuntu:~/Mycode/mosquitto-2.0.10.hard/mqtt_test$ gcc pub.c -o pub -lmosquitto
/usr/bin/ld: /usr/local/lib/libmosquitto.so: undefined reference to `dlopen'
/usr/bin/ld: /usr/local/lib/libmosquitto.so: undefined reference to `dlclose'
/usr/bin/ld: /usr/local/lib/libmosquitto.so: undefined reference to `dlerror'
/usr/bin/ld: /usr/local/lib/libmosquitto.so: undefined reference to `dlsym'
/usr/bin/ld: /usr/local/lib/libmosquitto.so: undefined reference to `dladdr'
collect2: error: ld returned 1 exit status
```

使用命令后添加 -ldl参数

> gcc pub.c -o pub -lmosquitto -ldl



订阅者会创建一个mqtt客户端，连接mqtt代理，订阅主题，然后等待相同主题的消息

- 连接上mqtt代理，执行回调函数`on_connect()`
- 订阅主题：topic1，执行回调函数`on_subscrib()`
- 有其他发布者发布了主题topic1的消息，执行回调函数`on_message()`
- 打印收到的消息

发布者会创建一个mqtt客户端，连接mqtt代理，然后等待主题的创建与发布

- 连接上mqtt代理，执行回调函数`my_connect_callback`
- 从终端读取消息，发布主题为topic1的消息，执行回调函数`my_publish_callback`

4.证书的请求与创建

4.1 openssl生成CA根证书

注意这里使用`./openssl`不能只是用`openssl`运行

- Step1：制作ca.key 私钥（aes256算法加密）

```
sudo ./openssl ecparam -genkey -name SM2 -out ./myCert/ca.key
```

  - -genkey——生成密钥
  - -name——指定生成密钥算法名称
  - -out——输出文件的路径
- Step2：制作生成CA证书请求

```
sudo ./openssl req -new -key ./myCert/ca.key -out ./myCert/ca.csr -subj
"/C=CN/ST=Hubei/L=Wuhan/O=HUST NCC/OU=NCLINK/CN=nclink.com/"
```

  - req——执行证书签发命令
  - -new——新证书签发请求
  - -key——指定私钥路径
  - -out——输出的csr文件的路径
  - -subj——证书相关的用户信息(subject的缩写)
  - 检查证书请求命令`openssl req -text -in ./myCert/ca.csr -noout`
- Step3：自签名得到根证书，注意这里也可以生成ca.crt文件。

```
cdq@cdq-ubuntu:/opt/local/bin$ sudo ./openssl x509 -req -days 3650 -sm3 -
extfile /opt/local/ssl/openssl.cnf -extensions v3_ca -signkey
./myCert/ca.key -in ./myCert/ca.csr -out ./myCert/ca.crt
Signature ok
subject=C = CN, ST = Hubei, L = Wuhan, O = HUST NCC, OU = NCLINK, CN =
nclink.com
Getting Private key
```

  - x509——生成x509格式证书
  - -req——输入csr文件
  - -days——证书的有效期（天）
  - -sm3——证书摘要采用sm3算法
  - -extfile——openssl配置文件路径
  - -extensions——按照openssl.cnf文件中配置的v3_ca项添加扩展
  - -signkey——签发证书的私钥
  - -in——要输入的csr文件
  - -out——输出的cer证书文件或者crt文件
  - 检查证书`./openssl x509 -text -in ./myCert/ca.cer -noout`
- 关于cer证书和crt证书相互转换

  CER是二进制形式的X.509证书，**DER**编码。
  CRT是二进制X.509证书，封装在文本（**base-64**）编码中。

  1. cer证书转crt证书：

```
sudo ./openssl x509 -inform PEM -in ./myCert/ca.cer -out ./myCert/ca.crt
```

  2. crt证书转cer证书

```
sudo ./openssl x509 -inform PEM -in ./myCert/ca.crt -out ./myCert/ca.cer
```

至此，CA根证书生成完毕，内容包含以下三个文件

- ca.key SM2算法私钥
- ca.csr　证书请求文件
- ca.crt　使用私钥签名后的CA根证书

4.2 签名证书和加密证书

1. 签名证书与加密证书

数字证书可分为签名证书和加密证书。

签名证书主要用于对用户信息进行签名，以保证信息的有效性和不可否认性；

加密证书主要用于对用户传送信息进行加密,以保证信息的保密性和完整性。

每个证书都包含一对密钥即签名公钥和签名私钥，加密公钥和加密私钥，将签名证书和加密证书的公钥公布于外。

签名时，用签名证书的私钥进行签名，其他用户可以利用公布于外网的签名公钥对签名进行验证。

加密时，用户B利用用户A公布于外网的加密公钥对信息进行加密传送给用户A，用户A利用自己的加密私钥对加密后的信

息进行解密得到完整的明文信息。

2. 数字证书格式

```
版本 ：该证书使用的是哪种版本的X.509标准
序列号：本项是CA分配给每一个证书的唯一的数字型编号，证书序列号的长度在不同的CA系统中是可配置的。
签名算法：本项用来指定颁发机构CA签发证书时使用的签名算法
颁发者：issuer 本项标识了颁发该证书的机构DN
有效期：validity 本项是证书的有效期和终止日期
主题：subject 证书拥有者的唯一是别名，这个字段必须是非空的。
有效的DN(Distinct Name)标识：
       c          country code (一般是c=cn)
       o          organization(组织)
       ou        organizational unit name(组织单位名)
       cn        common name (普通名)
       e          email (邮件地址)
       l           locality name (地址)
       st         state, or province name (国家或省份名)
       dc        domain Component (领域)
       uid      user id (用户标识符)
       t          title (标题)
       sn        device serial number name

公钥 public key: 本项用来标识公钥，公钥算法和公钥长度。
hash算法：和证书本身没多大关系，就是哈希算法，通常用MD5或SHA1.
hash值:是指对整个证书进行hash运算之后生成的一段数据，就是对证书的哈希摘要。
```

4.2 openssl生成服务器双证书

在上一步生成的CA根证书的基础上，为服务器`nclink.com`签名并颁发证书

在`./myCert`目录下创建文件夹`server`存放服务器的证书

1. 生成服务器的签名证书

- Step1：生成签名证书的私钥`SS_nclink.com.key`：

```
sudo ./openssl ecparam -genkey -name SM2 -out
./myCert/server/SS_nclink.com.key
```

- Step2：生成签名证书的证书请求文件`SS_nclin.com.csr`

```
sudo ./openssl req -new -key ./myCert/server/SS_nclink.com.key -out
./myCert/server/SS_nclink.com.csr -subj "/C=CN/ST=Hubei/L=Wuhan/O=HUST
NCC/OU=NCLINK/CN=nclink.com/"
```

- Step3：使用根CA证书签发nclink.com的签名证书`SS_nclink.com.crt`

```
sudo ./openssl x509 -req -days 3650 -sm3 -extfile /opt/local/ssl/openssl.cnf
-extensions v3_req -CA ./myCert/ca.crt -CAkey ./myCert/ca.key -CAserial
./myCert/ca.srl -CAcreateserial -in ./myCert/server/SS_nclink.com.csr -out
./myCert/server/SS_nclink.com.crt
```

- Step4：校验整数

```
/opt/local/bin/openssl verify -CAfile ./ca/ca.crt ./server/SS_nclink.com.crt
```

- 命令解释
  - -CA——指定CA证书的路径
  - -CAkey——指定CA证书的私钥路径
  - -CAserial——指定证书序列号文件的路径
  - -CAcreateserial——表示创建证书序列号文件(即上方提到的serial文件)，创建的序列号文件默认名称为-CA，指定的证书名称后加上.srl后缀，第一次创建证书需要使用该参数创建，此后不需要该参数

    > 在x509指令中，有多重方式可以指定一个将要生成证书的序列号，可以使用set_serial选项来直接指定证书的序列号，也可以使用-CAserial选项来指定一个包含序列号的文件。所谓的序列号是一个包含一个十六进制正整数的文件，在默认情况下，该文件的名称为输入的证书名称加上.srl后缀，比如输入的证书文件为ca.cer，那么指令会试图从ca.srl文件中获取序列号，可以自己创建一个ca.srl文件，也可以通过-CAcreateserial选项来生成一个序列号文件。

  - 注意：这里指定的-extensions的值为v3_req，在OpenSSL的配置中，v3_req配置的basicConstraints的值为CA:FALSE；而前面生成根证书时，使用的-extensions值为v3_ca，v3_ca中指定的basicConstraints的值为CA:TRUE，表示该证书是颁发给CA机构的证书。可以查看文件`/opt/local/ssl/openssl.cnf`

    ```
    [ v3_req ]
    # Extensions to add to a certificate request
    basicConstraints = CA:FALSE
    keyUsage = nonRepudiation, digitalSignature

    [ v3enc_req ]
    # Extensions to add to a certificate request
    basicConstraints = CA:FALSE
    keyUsage = keyAgreement, keyEncipherment, dataEncipherment

    [ v3_ca ]
    # Extensions for a typical CA
    # PKIX recommendation.
    subjectKeyIdentifier=hash
    authorityKeyIdentifier=keyid:always,issuer
    basicConstraints = critical,CA:true
    ```

- 在server文件夹下得到三个文件
  - `SS_nclink.com.key` 服务器签名证书的SM2私钥文件
  - `SS_nclink.com.csr` 服务器签名证书的 请求文件
  - `SS_nclink.com.crt` 服务器的签名证书（经过CA私钥签名认证）

2. 生成服务器的加密证书

- Step1：生成加密证书的私钥`SE_nclink.com.key`
- Step2：生成加密证书的证书请求文件`SE_nclink.com.csr`
- Step3：使用根证书CA签发服务器的加密证书`SE_nclink.com.crt`，**注意这里的`-extensions`参数使用的是`v3enc_req`**

```
sudo ./openssl ecparam -genkey -name SM2 -out
./myCert/server/SE_nclink.com.key

sudo ./openssl req -new -key ./myCert/server/SE_nclink.com.key -out
./myCert/server/SE_nclink.com.csr -subj "/C=CN/ST=Hubei/L=Wuhan/O=HUST
NCC/OU=NCLINK/CN=nclink.com/"

sudo ./openssl x509 -req -days 3650 -sm3 -extfile /opt/local/ssl/openssl.cnf
-extensions v3enc_req -CA ./myCert/ca.crt -CAkey ./myCert/ca.key -CAserial
./myCert/ca.srl -in ./myCert/server/SE_nclink.com.csr -out
./myCert/server/SE_nclink.com.crt
```

- 在server文件夹下增加了三个文件
  - `SE_nclink.com.key` 服务器签名证书的SM2私钥文件
  - `SE_nclink.com.csr` 服务器签名证书的 请求文件
  - `SE_nclink.com.crt` 服务器的签名证书（经过CA私钥签名认证）

4.3 openssl生成客户端双证书

假定生成客户端 admin的双证书，在`./myCert`文件夹下创建`client`文件夹，存放客户端证书文件，步骤和上面的一样，直接贴代码

1. 签名证书

```
sudo ./openssl ecparam -genkey -name SM2 -out ./myCert/client/SS_admin.key

sudo ./openssl req -new -key ./myCert/client/SS_admin.key -out
./myCert/client/SS_admin.csr -subj "/C=CN/ST=Hubei/L=Wuhan/O=HUST
NCC/OU=NCLINK/CN=admin/"

sudo ./openssl x509 -req -days 3650 -sm3 -extfile /opt/local/ssl/openssl.cnf -
extensions v3_req -CA ./myCert/ca.crt -CAkey ./myCert/ca.key -CAserial
./myCert/ca.srl -in ./myCert/client/SS_admin.csr -out
./myCert/client/SS_admin.crt
```

2. 加密证书

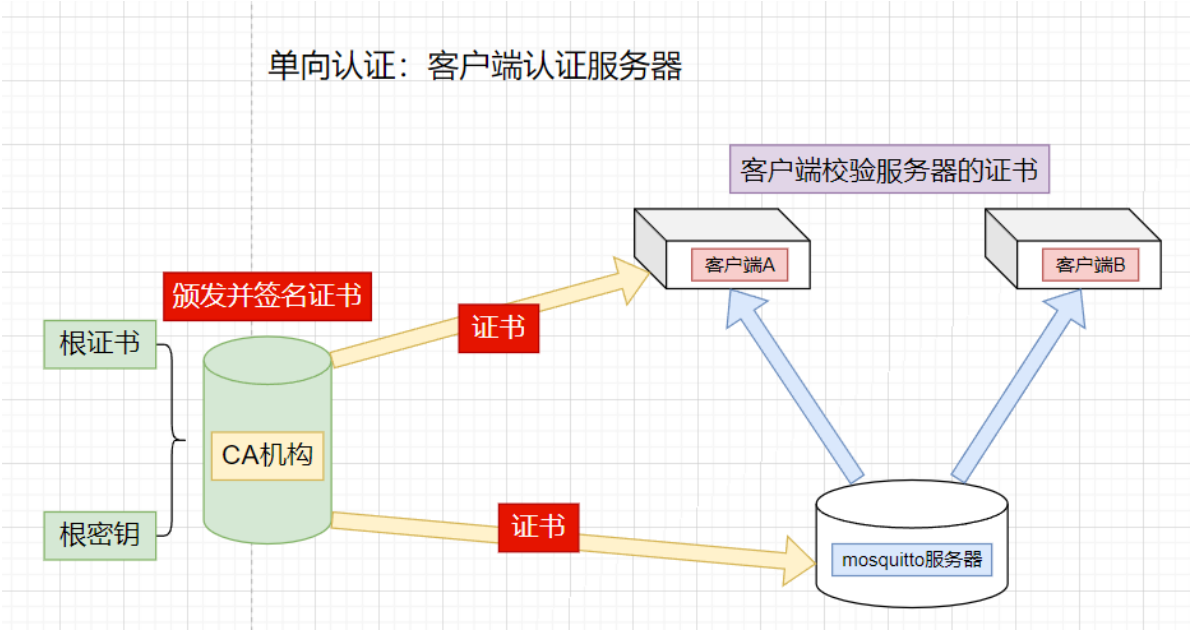
```
sudo ./openssl ecparam -genkey -name SM2 -out ./myCert/client/SE_admin.key

sudo ./openssl req -new -key ./myCert/client/SE_admin.key -out
./myCert/client/SE_admin.csr -subj "/C=CN/ST=Hubei/L=Wuhan/O=HUST
NCC/OU=NCLINK/CN=admin/"

sudo ./openssl x509 -req -days 3650 -sm3 -extfile /opt/local/ssl/openssl.cnf -
extensions v3enc_req -CA ./myCert/ca.crt -CAkey ./myCert/ca.key -CAserial
./myCert/ca.srl -in ./myCert/client/SE_admin.csr -out
./myCert/client/SE_admin.crt
```

5. 汤改版mosquitto单向认证

单向认证指的是客户端认证服务端。



5.1 修改mosquitto配置文件

修改文件`/etc/mosquitto/mosquitto.conf`

```
服务端口为8883端口：
listener 8883
#tls_engine tasscard_sm2
certfile /home/cdq/myCert/server/SS_nclink.com.crt
keyfile /home/cdq/myCert/server/SS_nclink.com.key
enc_certfile /home/cdq/myCert/server/SE_nclink.com.crt
enc_keyfile /home/cdq/myCert/server/SE_nclink.com.key
cafile /home/cdq/myCert/ca/ca.crt
require_certificate false
use_identity_as_username false
password_file /etc/mosquitto/pwfile.example
allow_anonymous false
```

重启`mosquitto`

`mosquitto -c /etc/mosquitto/mosquitto.conf`

5.2 单向认证测试

- 订阅端

```
mosquitto_sub -t test -h nclink.com -p 8883 -u user1 -P 123456 --cafile
/home/cdq/myCert/ca/ca.crt
SSL connection using GMTLSv1.1, (NONE)
无对端证书信息!
SSL connection using GMTLSv1.1, (NONE)
无对端证书信息!
SSL connection using GMTLSv1.1, ECC-SM4-SM3
对端证书信息:
证书: /C=CN/ST=Hubei/L=Wuhan/O=HUST NCC/OU=NCLINK/CN=nclink.com
颁发者: /C=CN/ST=BJ/L=Haidian/O=TASS/OU=DEV/CN=CA-common-
name/emailAddress=tassdev@tass.com.cn
SSL connection using GMTLSv1.1, ECC-SM4-SM3
对端证书信息:
证书: /C=CN/ST=Hubei/L=Wuhan/O=HUST NCC/OU=NCLINK/CN=nclink.com
```

```
颁发者: /C=CN/ST=BJ/L=Haidian/O=TASS/OU=DEV/CN=CA-common-
name/emailAddress=tassdev@tass.com.cn
hello
```

- 发布端

```
mosquitto_pub -t test -h nclink.com -p 8883 -u user1 -P 123456 -m "hello" --
cafile /home/cdq/myCert/ca/ca.crt
SSL connection using GMTLSv1.1, (NONE)
无对端证书信息!
SSL connection using GMTLSv1.1, (NONE)
无对端证书信息!
SSL connection using GMTLSv1.1, ECC-SM4-SM3
对端证书信息:
证书: /C=CN/ST=Hubei/L=Wuhan/O=HUST NCC/OU=NCLINK/CN=nclink.com
颁发者: /C=CN/ST=BJ/L=Haidian/O=TASS/OU=DEV/CN=CA-common-
name/emailAddress=tassdev@tass.com.cn
SSL connection using GMTLSv1.1, ECC-SM4-SM3
对端证书信息:
证书: /C=CN/ST=Hubei/L=Wuhan/O=HUST NCC/OU=NCLINK/CN=nclink.com
颁发者: /C=CN/ST=BJ/L=Haidian/O=TASS/OU=DEV/CN=CA-common-
name/emailAddress=tassdev@tass.com.cn
```

6. 汤改版mosquitto双向认证

6.1 修改mosquitto配置文件

修改文件`/etc/mosquitto/mosquitto.conf`

```
服务端口为8883端口:
listener 8884
#tls_engine tasscard_sm2
certfile /home/cdq/myCert/server/SS_nclink.com.crt
keyfile /home/cdq/myCert/server/SS_nclink.com.key
enc_certfile /home/cdq/myCert/server/SE_nclink.com.crt
enc_keyfile /home/cdq/myCert/server/SE_nclink.com.key
cafile /home/cdq/myCert/ca/ca.crt
require_certificate true
use_identity_as_username true
password_file /etc/mosquitto/pwfile.example
allow_anonymous false
```

重启`mosquitto`

`mosquitto -c /etc/mosquitto/mosquitto.conf`

5.2 双向认证测试

- 订阅端

```
mosquitto_sub -t test -h nclink.com -p 8884 -u user1 -P 123456 --cafile
/home/cdq/myCert/ca/ca.crt --cert /home/cdq/myCert/client/SS_admin.crt --key
/home/cdq/myCert/client/SS_admin.key --enc_cert
/home/cdq/myCert/client/SE_admin.crt --enc_key
/home/cdq/myCert/client/SE_admin.key
```

- 发布端

```
mosquitto_pub -t test -h nclink.com -p 8884 -u user1 -P 123456 -m "hello" --
cafile /home/cdq/myCert/ca/ca.crt --cert
/home/cdq/myCert/client/SS_admin.crt --key
/home/cdq/myCert/client/SS_admin.key --enc_cert
/home/cdq/myCert/client/SE_admin.crt --enc_key
/home/cdq/myCert/client/SE_admin.key
```

二、加密机配置

1. 静态ip设置

修改文件：`/etc/sysconfig/network-scripts/ifcfg-eno1`


```
# Generated by dracut initrd
NAME="eno1"
DEVICE="eno1"
ONBOOT=yes
NETBOOT=yes
UUID="1c8be5eb-361f-432a-ad86-80da6a4b687e"
IPV6INIT=no
BOOTPROTO=static
TYPE=Ethernet
IPADDR=192.168.19.19
NETMASK=255.255.255.0
GATEWAY=192.168.19.1
DNS1=114.114.114.114
```

注意：这里将ip设置成了192.169.19.19

2. 加密机配置汤改版mosquitto

```
|-root
|----work
|-------tassl
|-----------bin
|-----------include
|-----------lib
|-----------share
|-----------ssl
|-----------tassl_demo
|-------mosquitto
```

前提：gcc unzip Perl g++

```
yum install gcc
yum install unzip
yum install gcc-g++
```

新建`/root/work`目录，将`TASSL-1.1.1k-master.zip`和`mosquitto-2.0.10.hard.tar`复制到其目录下，解压。

```
tar -xvf mosquitto-2.0.10.hard.tar
unzip TASSL-1.1.1k-master.zip
```

2.1 安装TASSL

安装过程和上面类似，可能遇到的问题：

1. 缺少`Perl5`

```
[root@localhost TASSL-1.1.1k-master]# ./config --prefix=./ no-shared
Operating system: x86_64-whatever-linux2
You need Perl 5.
```

解决办法：下载安装wget，这里安装到`~/work/localPerl`

```
    yum install wget
    wget https://www.cpan.org/src/5.0/perl-5.30.1.tar.gz
    tar -xzf perl-5.30.1.tar.gz
    cd perl-5.30.1
    ./Configure -des -Dprefix=~/work/localPerl
    make
    make test
    make install
```

2. 执行配置文件`./config --prefix=/root/work/tassl no-shared`
3. 安装 make & makeinstall

2.2 安装汤改版mosquitto

1. 修改配置文件`config.mk`的tassl路径
2. 安装`make & make install`
3. 修改mosquitto配置文件`/etc/mosquitto/mosquitto.conf`
4. 运行mosquitto`mosquitto -c /etc/mosquitto/mosquitto.conf  -d`

   ```
   [root@localhost mosquitto]# mosquitto -c /etc/mosquitto/mosquitto.conf  -d
   1656507238: Warning: Unable to drop privileges to 'mosquitto' because this
   user does not exist. Trying 'nobody' instead.
   ```

   这里可能会报错，解决办法：需要给系统创建mosquitto用户和组

   ```
   groupadd mosquitto
   useradd -g mosquitto mosquitto
   chown -R mosquitto:mosquitto /etc/mosquitto/
   ```

5. 测试mosquitto `mosquitto_sub -t test`

   这里可能会报错

   ```
   mosquitto_sub -t test
   mosquitto_sub: error while loading shared libraries: libmosquitto.so.1:
   cannot open shared object file: No such file or directory
   ```

   可能是这个库目录并没有加入到该环境变量中，解决方案：

   ```
   # 将该目录加入到共享库的配置文件中
   echo "/usr/local/lib" >> /etc/ld.so.conf
   ldconfig
   ```

   继续报错

   ```
   [root@localhost mosquitto]# mosquitto_pub -t test -m "hello"
   Connection error: Connection Refused: not authorised.
   ```

   这个错误是身份认证的问题，使用mosquitto_passwd添加用户`mosquitto_passwd -c /etc/mosquitto/pwfile.example user1`，密码`123456`.

6. 带用户名密码的mosquitto测试通过`mosquitto_pub -t test -m "hello" -u user1 -P 123456`

   另一端订阅此消息，能正常收到消息。注意这里的P是大写，小写p用于指定端口号，大写P指定密码

   ```
   [root@localhost ~]# mosquitto_sub -t test -u user1 -P 123456
   hello
   ```

至此，汤改版mosquitto在加密机上配置完毕。之后的单向认证和双向认证参考内容一，mosquitto 汤改版学习。