openssl 制作证书过程

一. 生成CA证书

1. 创建 1024位 私钥 至 ca目录
    - 1. `openssl genrsa -out ca/ca-key.pem 1024`
2. 根据上步 *私钥* 创建证书请求文件至 ca 目录
    - 1. `openssl req -new -out ca/ca-req.csr -key ca/ca-key.pem`
    - 输入信息分别为: cn, beijing, haidian, nanhao, nanhao, nanhao, key,
3. 自签署证书:(根据上两步生成的*私钥* 和*申请文件*创建 自签署证书至 ca 目录, 至此已完成公私钥生成)
    - `openssl x509 -req -in ca/ca-req.csr -out ca/ca-cert.pem -signkey ca/ca-key.pem -days 3650`
4. 将自签署证书导出为浏览器支持的 .p12格式(不需要的可以省略)(根据上几步生成的私钥与自签署证书将其 格式转换为 .p12)
    - `openssl pkcs12 -export -clcerts -in ca/ca-cert.pem -inkey ca/ca-key.pem -out ca/ca.p12`
    - 密码 : nanhao (我也忘了)

二. 生成服务器证书

1. 创建 1024位 私钥 至 server 目录:
    - `openssl genrsa -out server/server-key.pem 1024`
2. 根据上步 *私钥* 创建证书请求文件至 server 目录
    - `openssl req -new -out server/server-req.csr -key server/server-key.pem`
    - 输入信息分别为 : cn, beijing, chaoyang, baidu, baidu, 123456@qq.com, test,(注意: 次逗号只是起书面分隔作用, 实际输入时并没有)
3. 服务器自签署证书(根据上两步生成的 服务器证书请求文件与私钥, 加 生成的CA 认证中心的自签署证书(.pem格式的)和CA 私钥 生成日期为 3650天的 服务器自签署证书)
    - `openssl x509 -req -in server/server-req.csr -out server/server-cert.pem -signkey server/server-key.pem -CA ca/ca-cert.pem -CAkey ca/ca-key.pem -CAcreateserial -days 3650`
4. 将服务器自签署证书导出为 浏览器支持的格式 .p12(根据生成的 服务器私钥与服务器自签署证书将其 格式转换为 .p12)
    - `openssl pkcs12 -export -clcerts -in server/server-cert.pem -inkey server/server-key.pem -out server/server.p12`
    - 密码 : 123456

三. 生产client证书

1. 创建 1024位 私钥 至 server 目录:
    - `openssl genrsa -out client/client-key.pem 1024`
2. 根据上步 *私钥* 创建证书请求文件至 client 目录
    - `openssl req -new -out client/client-req.csr -key client/client-key.pem`
    - 输入信息 : cn, beijing, haidian, xiaomi, xiaomi, xiaomi, test, 123456, xiaomi,

3. 客户端自签署证书(根据上两步生成的 客户端证书请求文件与私钥, 加 生成的CA 认证中心的自签署证书(.pem格式的)和CA 私钥 生成日期为 3650天的 服务器自签署证书)
    - `openssl x509 -req -in client/client-req.csr -out client/client-cert.pem -signkey client/client-key.pem -CA ca/ca-cert.pem -CAkey ca/ca-key.pem -CAcreateserial -days 3650`
4. 将客户端自签署证书导出为 浏览器支持的格式 .p12(根据生成的客户端私钥与客户端自签署证书将其 格式转换为 .p12)
    - `openssl pkcs12 -export -clcerts -in client/client-cert.pem -inkey client/client-key.pem -out client/client.p12`
    - 密码 : 123456

四. 根据ca证书生成 jks文件

1. jks是JAVA的keytools证书工具支持的证书私钥格式。
2. `keytool -keystore C:/openssl/bin/jks/truststore.jks -keypass 222222 -storepass 222222 -alias ca -import -trustcacerts -file C:/openssl/bin/ca/ca-cert.pem`

五. 配置 Tomcat 服务器

1. 本次实验环境为 Tomcat 8
2. 配置文件修改的内容为 :

```
<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"
maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS"
keystoreFile="/home/yyx/certificate/server/server-cert.p1
keystorePass="123456" keystoreType="PKCS12"
truststoreFile="/home/yyx/certificate/jks/truststore.jks"
truststorePass="222222" truststoreType="JKS" />
```

六. 导入证书

1. 将ca.p12，client.p12分别导入到IE中去（打开IE->;Internet选项->内容->证书）。
2. ca.p12导入至受信任的根证书颁发机构，client.p12导入至个人

七. OpenSSL安全套接字通信

1. *安全连接要求在连接建立后进行握手*。在握手过程中，服务器向客户机发送一个证书， 然后，客户机根据一组可信任证书来核实该证书。它还将检查证书，以确保它没有过期。要 *检验证书是可信任*的，需要在*连接建立之前*提前*加载一个 可信任证书库*.
2. 只有在服务器发出请求时，客户机才会向服务器发送一个证书。该过程叫做*客户机认证*。使用证书， 在客户机和服务器之间传递密码参数，以建立安全连接。尽管握手是在建立连接之后才进行的，但是客户机或服务器可以在任何时刻请求进行一次新的握手。

八. 直接从私钥中 提取公钥信息

1. 本小节, 不会生成证书, 注意: *证书与公钥信息不同*. 证书包含公钥信息 和 其他认证信息.

2. 生成私钥
    - openssl genrsa -out plainPrv.key 1024
3. 直接生成公钥
    - 第一种提取方式 : openssl rsa -in plainPrv.key -pubout -out plainPub.key
    - 这种方法提取出的公钥 必须采用该函数方法 提取信息 : `PEM_read_RSA_PUBKEY()`
    - 第二种提取方式 : openssl rsa -in plainPrv.key -RSAPublicKey_out -out plainPub.key
    - 这种方法提取出的公钥 必须采用该函数方法 提取信息 : `PEM_read_RSA_PUBKEY()`

九. OpenSSL API 加密信息

`int RSA_public_encrypt(int flen, const unsigned char *from,unsigned char *to, RSA *rsa,int padding);` 该方法 最大加密数据长度为 : 245 字节.