SSL配置全过程

生成证书可以通过两种方式：openssl(X509)和keytool(jks)。
本文介绍的是keytool的方式。 （暂且不谈两者的优劣）

首先，介绍下几个文件的含义。
①12580_com.key 私钥文件，自己保留！
②12580_com.csr 申请证书所需要的文件，用于提交给证书机构。
③12580_com.jks 密钥库。
④temp.p12和temp.pem 从密钥库中导出密钥过程中涉及到的中间文件。
⑤12580_com.pem 从证书厂商处获得的证书文件（不一定直接提供现成的文件给我们！）
其中，前四个都是我们自己服务器上生成的，最后一个是由证书厂商提供。

下面是前期准备的具体步骤。
①生成密钥库

```
keytool -genkey -alias 12580_com -keyalg RSA -sigalg SHA1withRSA -keysize 2048 -keystore 12580_com.jks -storepass ca.12580.com -keypass ca.12580.com
```

```
您的名字与姓氏是什么?
  [Unknown]:  *.12580.com
您的组织单位名称是什么?
  [Unknown]:  E-commerce Operations Center
您的组织名称是什么?
  [Unknown]:  China Mobile Group Jiangsu Co., Ltd.
您所在的城市或区域名称是什么?
  [Unknown]:  Suzhou
您所在的省/市/自治区名称是什么?
  [Unknown]:  Jiangsu
该单位的双字母国家/地区代码是什么?
  [Unknown]:  CN
```

-alias:密钥库中可包含多个密钥串，该参数为密钥串的别名，下面会用到，不能记错。需要唯一。
-keysize:根据移动邮件中提到的要求，密钥长度选择2048。
-keystore:密钥库的名字，随意。
-storepass:密钥库的密码。
-keypass:私钥的密码。可以和密钥库的密码不一样，这里只是为了方便而已。
另外，第二个方框中的信息，由移动提供。

②生成csr文件，提供给证书厂商（这里我们是交由 移动-岳威 提供给厂商）

```
keytool -certreq -alias 12580_com -sigalg SHA1withRSA -file 12580_com.csr -keystore12580_com.jks -keypass ca.12580.com -storepass ca.12580.com
```

-alias:需要使用的密钥串的别名。就是①里面那个。
-sigalg:签名加密算法：SHA1withRSA和SHA256withRSA。 这里选择正常的SHA1。将来必定是要选择SHA256的。
-file:生成的csr文件。
-keystore:指定密钥库。
-keypass:私钥密码。
-storepass:密钥库密码。

③从密钥库中的密钥串里将我们所需的私钥提取出来

提取方式有两种：一、通过java代码（不适合我们）；二、先转成pkcs12文件，再转成pem文件，见下。

keytool -importkeystore -srckeystore 12580_com.jks -destkeystore temp.p12 -srcstoretype JKS -deststoretype PKCS12 -srcstorepass ca.12580.com -deststorepass ca.12580.com
openssl pkcs12 -in temp.p12 -out temp.pem -passin pass:ca.12580.com -passout pass:ca.12580.com
复制temp.pem文件中BEGIN RSA PRIVATE KEY(包含)到END RSA PRIVATE KEY(包含)一行的内容到新文件12580_com.key，该文件就是我们所需的私钥文件。

下面是在拿到证书厂商提供的证书之后的具体步骤。

①获取证书文件

以本次申请证书为例，在证书厂商的回复邮件中三个文件：一个是web服务器证书，两个是中级CA证书（其实我也不知道什么意思）

直接将他们以文本的形式复制到文件中，注意：复制的顺序要先是web服务器证书，而后是两个中级CA证书。

然后将该文本重命名成**12580_com.pem**。 这就是我们要的证书文件。

②获取私钥文件

私钥文件我们在上面的准备工作中已经生成了，就是**12580_com.key**

所有文件准备齐全了，可以进行nginx配置。

```
server {
        listen       443;
        server_name  ca.12580.com;
        ssl                  on;
        ssl_certificate      12580_com/12580_com.pem;
        ssl_certificate_key  12580_com/12580_com.key;
        ssl_session_timeout  5m;
        ssl_protocols  SSLv3 TLSv1;
        ssl_ciphers  HIGH:!ADH:!EXPORT56:RC4+RSA:+MEDIUM;
        ssl_prefer_server_ciphers   on;

        ……
        ……
        ……
}
```

在两句标蓝的话里面指定下上面获得的证书文件和私钥文件即可。

注意：目录是在nginx/conf之下的。

至此，整个ssl配置已经完成。已经可以成功访问https://ca.12580.com了。

下面再说说在配置完成之后发生的两个问题。

①配置nginx支持多域名ssl

在刚配置完之后访问ca.12580.com的发现浏览器显示证书依旧是12580life.com的，于是就想到可能是nginx默认不支持多域名ssl。

解决该问题就两步：
一、下载openssl包（虽然系统已经有openssl，但是还是下载一下新的）

```
wget http://www.openssl.org/source/openssl-1.0.1c.tar.gz
```

二、重新编辑nginx，加上下面一句话（路径就是上面下载下来的openssl包的路径）

```
--with-openssl=/usr/local/src/openssl-1.0.1c
```

完成之后能够发现 ./nginx -V的结果发生变化，从原来的

```
TLS SNI support disabled
```

变成了

```
TLS SNI support enabled
```

②配置nginx重启不需要密码
加了ssl之后，每次./nginx -t和./nginx reload的时候都需要输入密码，是因为我们私钥文件中加入了密码。将密码从私钥文件中去除即可解决该问题。

```
openssl rsa -in 12580_com.key -out 12580_com.key.nopwd
```

新生成的12580_com.key.nopwd就是不带密码的私钥文件（命名随意）。
最后在nginx配置文件中修改下配置即可

```
        ssl_certificate      12580_com/12580_com.pem;
        ssl_certificate_key  12580_com/12580_com.key.nopwd;
```

发现重启就不需要密码了。