**OpenApi只需要产生genkey即可**



**1.创建客户端keystore文件**

[?](http://my.oschina.net/xinxingegeya/blog/266826)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | E:\javassl>keytool -genkey -alias sslclient -keystore sslclientkeys  输入密钥库口令:  再次输入新口令:  您的名字与姓氏是什么?    [Unknown]:  client  您的组织单位名称是什么?    [Unknown]:  dev  您的组织名称是什么?    [Unknown]:  rd  您所在的城市或区域名称是什么?    [Unknown]:  sy  您所在的省/市/自治区名称是什么?    [Unknown]:  sy  该单位的双字母国家/地区代码是什么?    [Unknown]:  cn  CN=client, OU=dev, O=rd, L=sy, ST=sy, C=cn是否正确?    [否]:  y    输入 <sslclient> 的密钥口令          (如果和密钥库口令相同, 按回车):    E:\javassl> |

**2.将客户端keystore文件导出证书格式**

[?](http://my.oschina.net/xinxingegeya/blog/266826)

|  |  |
| --- | --- |
| 1  2  3 | E:\javassl>keytool -export -alias sslclient -keystore sslclientkeys -file sslclient.cer  输入密钥库口令:  存储在文件 <sslclient.cer> 中的证书 |

**3.创建一个服务器端 keystore 文件**

[?](http://my.oschina.net/xinxingegeya/blog/266826)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | E:\javassl>keytool -genkey -alias sslserver -keystore sslserverkeys  输入密钥库口令:  再次输入新口令:  您的名字与姓氏是什么?    [Unknown]:  lyx  您的组织单位名称是什么?    [Unknown]:  rj  您的组织名称是什么?    [Unknown]:  gc  您所在的城市或区域名称是什么?    [Unknown]:  cn  您所在的省/市/自治区名称是什么?    [Unknown]:  cn  该单位的双字母国家/地区代码是什么?    [Unknown]:  cn  CN=lyx, OU=rj, O=gc, L=cn, ST=cn, C=cn是否正确?    [否]:  y    输入 <sslserver> 的密钥口令          (如果和密钥库口令相同, 按回车): |

**4.将服务器端 keystore 文件导出成证书格式**

[?](http://my.oschina.net/xinxingegeya/blog/266826)

|  |  |
| --- | --- |
| 1  2  3 | E:\javassl>keytool -export -alias sslserver -keystore sslserverkeys -file sslserver.cer  输入密钥库口令:  存储在文件 <sslserver.cer> 中的证书 |

**5.将客户端证书导入到服务器端受信任的 keystore 中**

[?](http://my.oschina.net/xinxingegeya/blog/266826)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | E:\javassl>keytool -import -alias sslclient -keystore sslservertrust -file sslclient.cer  输入密钥库口令:  再次输入新口令:  所有者: CN=client, OU=dev, O=rd, L=sy, ST=sy, C=cn  发布者: CN=client, OU=dev, O=rd, L=sy, ST=sy, C=cn  序列号: 23a77f2b  有效期开始日期: Tue May 20 12:18:19 CST 2014, 截止日期: Mon Aug 18 12:18:19 CST 2014  证书指纹:           MD5: E6:4E:2D:67:5F:E3:46:36:EB:A0:69:8A:79:BE:92:9B           SHA1: 8F:40:AC:71:84:3A:03:64:63:22:E6:29:91:DD:77:9B:B9:E2:F1:C6           SHA256: D9:DB:0D:86:44:1B:06:B2:89:29:C1:6A:84:50:C0:A7:F2:BD:A1:DD:11:EC:54:F5:50:AF:11:73  :C9:DE:D3:37           签名算法名称: SHA1withDSA           版本: 3    扩展:    #1: ObjectId: 2.5.29.14 Criticality=false  SubjectKeyIdentifier [  KeyIdentifier [  0000: 75 31 80 52 AA C7 6B 36   3E E5 73 23 3F 2D 67 54  u1.R..k6>.s#?-gT  0010: 0D 60 4D 18                                        .`M.  ]  ]    是否信任此证书? [no]:  y  证书已添加到密钥库中 |

**6.将服务器端证书导入到客户端受信任的 keystore 中**

注：由于服务端的证书是我们自己生成的，没有任何受信任机构的签名，所以客户端是无法验证服务端证书的有效性的，通信必然会失败。所以我们需要为客户端创建一个保存所有信任证书的仓库，然后把服务端证书导进这个仓库。这样，当客户端连接服务端时，会发现服务端的证书在自己的信任列表中，就可以正常通信了。

这就是trustStore的作用。

同理，服务器端的trustStore也是一样。

[?](http://my.oschina.net/xinxingegeya/blog/266826)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | E:\javassl>keytool -import -alias sslserver -keystore sslclienttrust -file sslserver.cer  输入密钥库口令:  再次输入新口令:  所有者: CN=lyx, OU=rj, O=gc, L=cn, ST=cn, C=cn  发布者: CN=lyx, OU=rj, O=gc, L=cn, ST=cn, C=cn  序列号: 2cbfdf00  有效期开始日期: Tue May 20 12:25:00 CST 2014, 截止日期: Mon Aug 18 12:25:00 CST 2014  证书指纹:           MD5: 83:30:0D:45:F5:45:7D:D8:E7:4C:56:99:A1:08:C8:C6           SHA1: B1:C0:D8:AE:CA:5E:C0:16:14:28:4E:75:6A:09:80:BC:9D:90:E5:E8           SHA256: 3E:73:A5:3C:54:EC:3C:79:20:23:7A:82:70:83:B2:5B:2A:99:0E:52:B8:0B:9C:E5:8A:03:B2:10  :9C:74:90:00           签名算法名称: SHA1withDSA           版本: 3    扩展:    #1: ObjectId: 2.5.29.14 Criticality=false  SubjectKeyIdentifier [  KeyIdentifier [  0000: F3 1F DE CA 36 41 A2 5A   F5 A1 1F 4E 98 DF 7A 04  ....6A.Z...N..z.  0010: 8B 13 1B 23                                        ...#  ]  ]    是否信任此证书? [no]:  y  证书已添加到密钥库中    E:\javassl> |

**7.查看密钥库**

[?](http://my.oschina.net/xinxingegeya/blog/266826)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | E:\javassl>keytool -list -keystore sslclienttrust  输入密钥库口令:    密钥库类型: JKS  密钥库提供方: SUN    您的密钥库包含 1 个条目    sslserver, 2014-5-20, trustedCertEntry,  证书指纹 (SHA1): B1:C0:D8:AE:CA:5E:C0:16:14:28:4E:75:6A:09:80:BC:9D:90:E5:E8    E:\javassl> |

准备工作完成，运行SSL的客户端和服务器端