[Java keytool工具的作用及使用方法](http://blog.csdn.net/ocelight/article/details/5775700)

2010-07-30 10:16 349人阅读 [评论](http://blog.csdn.net/ocelight/article/details/5775700#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/ocelight/article/details/5775700#report)

[java](http://www.csdn.net/tag/java)[工具](http://www.csdn.net/tag/%e5%b7%a5%e5%85%b7)[算法](http://www.csdn.net/tag/%e7%ae%97%e6%b3%95)[加密](http://www.csdn.net/tag/%e5%8a%a0%e5%af%86)[存储](http://www.csdn.net/tag/%e5%ad%98%e5%82%a8)[archive](http://www.csdn.net/tag/archive)

Keytool 是安全钥匙与证书的管理工具.它管理一个存储了私有钥匙和验证相应公共钥匙的与它们相关联的X.509   
证书链的keystore(相当一个数据库).   
Keytool 是一个有效的安全钥匙和证书的管理工具.   
它能够使用户使用数字签名来管理他们自己的私有/公共钥匙对,管理用来作自我鉴定的相关的证书,管理数据完整性和鉴定服务.它还能使用户在通信时缓存它们  
的公共钥匙.   
一个证书是某一实体(个人,公司等)的数字签名,指出其他实体的公共钥匙(或其他信息)的详细的值.当数据被签名后,这个签名信息被用来检验数据的完整性  
和真实性.完整性指数据没有被修改和篡改,真实性指数据从任何产生和签名的一方真正的传输到达.   
Keytool 把钥匙和证书储存到一个keystore.默任的实现keystore的是一个文件.它用一个密码保护钥匙.   
而另外的一个工具jarsigner用keystore中的信息产生或检验Java aRchive(jar文件)中的数字签名.   
Keystore有两个不同的入口:   
1.钥匙入口:保存了非常敏感的加密的钥匙信息,并且是用一个保护的格式存储以防止未被授权的访问.以这种形式存储的钥匙是秘密钥匙,或是一个对应证书链  
中公有钥匙的私有钥匙.   
2.信任证书入口:包含一个属于其他部分的单一公共钥匙证书.它之所以被称为&quot;信任证书&quot;,是因为keystore信任的证书中的公共钥匙真正属于证书所  
有者的身份识别.   
Keystore的别名:   
所有的keystore入口(钥匙和信任证书入口)是通过唯一的别名访问.别名是   
不区分大小写的.如别名Hugo和hugo指向同一个keystore入口.   
可以在加一个入口到keystore的时候使用-genkey参数来产生一个钥匙对(公共钥匙和私有钥匙)时指定别名.也可以用-import参数加一个  
证书或证书链到信任证书.   
如:   
keytool -genkey -alias duke -keypass dukekeypasswd   
其中duke为别名,dukekeypasswd为duke别名的密码.这行命令的作用是产生一个新的公共/私有钥匙对.   
假如你想修改密码,可以用:   
keytool -keypasswd -alias duke -keypass dukekeypasswd -new newpass   
将旧密码dukekeypasswd改为newpass.   
   
Keystore的产生:   
1.当使用-genkey   
或-import或-identitydb命令添加数据到一个keystore,而当这个keystore不存在时,产生一个keystore.默认名  
是.keystore,存放到user-home目录.   
2.当用-keystore指定时,将产生指定的keystore.   
Keystore的实现:   
Keytool 类位于java.security包下,提供一个非常好的接口去取得和修改一个keystore中的信息.   
目前有两个命令行:keytool和jarsinger,一个GUI工具Policy   
可以实现keystore.由于keystore是公开的,用户可以用它写一些额外的安全应用程序.   
Keystore还有一个sun公司提供的內在实现.它把keystore作为一个文件来实现.利用了一个keystore类型(格式)&quot;JKS&quot;.它用  
单独的密码保护每一个私有钥匙.也用可能不同的密码保护整个keystore的完整性.   
支持的算法和钥匙大小:   
keytool允许用户指定钥匙对和注册密码服务供应者所提供的签名算法.缺省的钥匙对产生算法是&quot;DSA&quot;.假如私有钥匙是&quot;DSA&quot;类型,缺省签名算  
法是&quot;SHA1withDSA&quot;,假如私有钥匙是&quot;RSA&quot;类型,缺省算法是&quot;MD5withRSA&quot;.   
当产生一个DSA钥匙对,钥匙必须在512-1024位之间.对任何算法的缺省钥匙大小是1024位.   
证书:   
一个证书是一个实体的数字签名,指出其他实体的公共钥匙有明确的值.   
1.公共钥匙 :是同一个详细的实体的数字关联,并有意让所有想同这个实体发生信任关系的其他实体知道.公共钥匙用来检验签名;   
2.数字签名:假如数据已被签名,并用身份存储在一个实体中,一个签名能够证明这个实体知道这个数据.这个数据用实体私有钥匙签名并递交;   
3.身份:知道实体的方法.在一些系统中身份是公共钥匙,其他系统中可以是从一个X.509名字的邮件地址的Unix UID来的任何东西;   
4.签名:一个签名用用实体私有钥匙来计算某些加密数据;   
5.私有钥匙:是一些数字,每一个私有钥匙只能被特定的拥有该私有钥匙的实体知道.私有和公共钥匙存在所有用公共钥匙加密的系统的钥匙对中.一个公共钥匙  
加密(如DSA),一个私有钥匙与一个正确的公共钥匙通信.私有钥匙用来计算签名.   
6.实体:一个实体可以是一个人,一个组织,一个程序,一台计算机,一个商业,一个银行,或其他你想信任的东西.   
   
Keytool应用实例:   
1.产生一个keystore:   
keytool -genkey -alias User(keystore的别名) -keyalg RSA -validity 7   
-keystore keystore(指定keystore).   
运行这个命令,系统提示:   
Enter keystore password:yourpassword(输入密码)   
What is your first and last name?   
[Unknown]: your name(输入你的名字)   
What is the name of your organizational unit?   
[Unknown]:your organizational(输入你所在组织单位的名字)   
What is the name of your organization?   
[Unknown]:your organization name (输入你所在组织的名字)   
What is the name of your City or Locality?   
[Unknown]:your city name(输入所在城市的名字)   
What is the name of your State or Province?   
[Unknown]:your provice name(输入所在省份名字)   
What is the two-letter country code for this unit?   
[Unknown]:cn(输入国家名字)   
Is CN=your name, OU=your organizaion, O=&quot;your organization name&quot;,   
L=your city name, ST=your province name, C=cn correct?   
[no]: yes   
   
2.检查一个keystore:   
keytool -list -v -keystore keystore   
Enter keystore password:your password(输入密码)   
将显示keystore內容如:   
Keystore type: jks   
Keystore provider: SUN   
   
Your keystore contains 1 entry   
   
Alias name: yourname   
Creation date: Dec 20, 2001   
Entry type: keyEntry   
Certificate chain length: 1   
Certificate[1]:   
Owner: CN=yourname, OU=your organization, O=&quot;your organization name&quot;,   
L=your city name, ST=your province name, C=CN   
Issuer: CN=Duke, OU=Java Software, O=&quot;Sun Microsystems, Inc.&quot;, L=Palo   
Alto, ST=CA, C=US   
Serial number: 3c22adc1   
Valid from: Thu Dec 20 19:34:25 PST 2001 until: Thu Dec 27 19:34:25 PST   
2001   
Certificate fingerprints:   
MD5: F1:5B:9B:A1:F7:16:CF:25:CF:F4:FF:35:3F:4C:9C:F0   
SHA1: B2:00:50:DD:B6:CC:35:66:21:45:0F:96:AA:AF:6A:3D:E4:03:7C:74   
3.输出keystore到一个文件:testkey:   
keytool -export -alias duke -keystore keystore -rfc -file testkey   
系统输出:   
Enter keystore password:your password(输入密码)   
Certificate stored in file   
4.输入证书到一个新的truststore:   
keytool -import -alias dukecert -file testkey -keystore truststore   
Enter keystore password:your new password.(输入truststore新密码)   
   
5.检查truststore:   
keytool -list -v -keystore truststore   
系统将显示truststore的信息.   
现在可以用适当的keystore运行你的应用程序.如:   
java -Djavax.net.ssl.keyStore=keystore   
-Djavax.net.ssl.keyStorePassword=password Server   
和: java -Djavax.net.ssl.trustStore=truststore   
-Djavax.net.ssl.trustStorePassword=trustword Client