

公告



可以扫文章下面的二维码联系我哟

昵称：救火队长
园龄：2年11个月
粉丝：126
关注：8
[+加关注](#)

<	2016年3月						>
日	一	二	三	四	五	六	
28	29	1	2	3	4	5	
6	7	8	9	10	11	12	
13	14	15	16	17	18	19	
20	<u>21</u>	22	<u>23</u>	24	25	26	
27	28	29	30	31	1	2	
3	4	5	6	7	8	9	

随笔-123 文章-1 评论-125

Java Security : Java加密框架(JCA)简要说明

加密服务总是关联到一个特定的算法或类型,它既提供了密码操作(如Digital Signature或MessageDigest),生成或供应所需的加密材料(Key或Parameters)加密操作,也会以一个安全的方式生成数据对象(KeyStore或Certificate),封装(压缩)密钥(可以用于加密操作)。

Java Security API中,一个engine class就是定义了一种加密服务,不同的engine class提供不同的服务。下面就来看看有哪些engine class :

- 1) MessageDigest : 对消息进行hash算法生成消息摘要 (digest) 。
- 2) Signature : 对数据进行签名、验证数字签名。
- 3) KeyPairGenerator : 根据指定的算法生成配对的公钥、私钥。
- 4) KeyFactory : 根据Key说明 (KeySpec) 生成公钥或者私钥。
- 5) CertificateFactory : 创建公钥证书和证书吊销列表 (CRLs) 。
- 6) KeyStore : keystore是一个keys的数据库。Keystore中的私钥会有一个相关联的证书链,证书用于鉴定对应的公钥。一个keystore也包含其它的信任的实体。
- 7) AlgorithmParameters : 管理算法参数。KeyPairGenerator就是使用算法参数,进行算法相关的运算,生成KeyPair的。生成Signature时也会用到。
- 8) AlgorithmParametersGenerator : 用于生成AlgorithmParameters 。

搜索

找找看

常用链接

[我的随笔](#)

[我的评论](#)

[我的参与](#)

[最新评论](#)

[我的标签](#)

最新随笔

[1. Jenkins 快速开始——自动化构建](#)

[2. Checkstyle：整洁你的代码](#)

[3. Maven Lifecycle](#)

[4. JDBC API Description](#)

[5. JDBC Driver Types](#)

[6. Tomcat：利用Apache配置反向代理、负载均衡](#)

[7. SLF4j：Log facade abstract](#)

[8. IO redirect](#)

[9. Java：Remote Debug](#)

[10. Thrift：Quick Start](#)

我的标签

[java\(12\)](#)

[Spring\(12\)](#)

[tomcat\(12\)](#)

[Cache\(5\)](#)

9) SecureRandom：用于生成随机数或者伪随机数。

10) CertPathBuilder：用于构建证书链。

11) CertPathValidator：用于校证书链。

12) CertStore：存储、获取证书链、CRLs到（从）CertStore中。

从上面这些engine class中，可以看出JCA（Java加密框架）中主要就是提供了4种服务：Digest、Key、Cert、Signature、Algorithm。

1) 对消息内容使用某种hash算法就可以生成Digest。

2) 利用KeyFactory、KeyPairGenerator就可以生成公钥、私钥。

3) 证书中心使用公钥就可生成Cert。

4) 可以使用私钥和Digest就可以消息进行签名Signature。

5) 不论是Digest、Key、Cert、Signature，都要使用到算法Algorithm。

JCA Core API

1) engine class的提供商Provider

从JCA的设计上来说，这些engine的实现都离不开Provider。

这个类继承了Properties，提供了JCA中的engine class。每个engine class都有

Hibernate(4)
eclipse(4)
aop(4)
Java Security(4)
nio(4)
Memcached(3)
更多

随笔分类 (179)

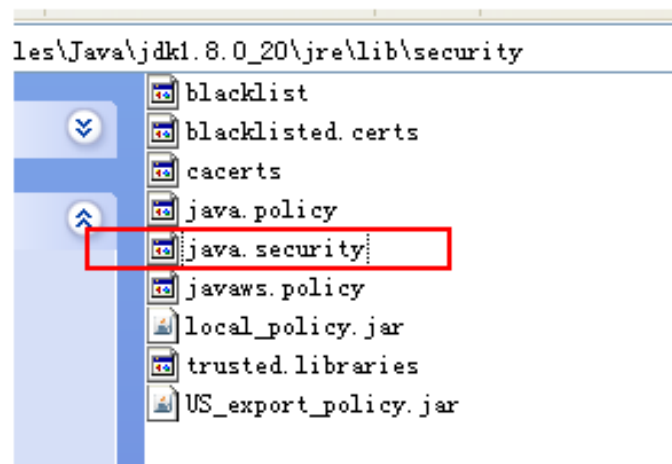
Android(3)
AOP(5)
Apache Commons(2)
C、C++(3)
Cache(7)
CAS(2)
Git(3)
GlassFish(1)
Gradle
Hibernate(3)
HTML、HTML5、CSS(4)
J2EE(11)
J2SE(33)
Java XML(2)
JavaScript(8)
jQuery(4)
JTA(1)
Linux(5)

getInstance()方法，它们都是从provider中获取相关实例的。所以说Provider是JCA engine class的提供商。

2) 管理Provider的工具：Security

其实就是一个存放Provider的集合。如果你自定义了一个Provider，可以使用Java Security属性文件配置provider，也可以直接使用Security采用编程的方式来添加Provider。然后就可以使用自定义的engine class了。

Java Security 属性文件在Java Security Policy中已有提过。在安装目录下：



下面是一个自定义的Provider：



Logger(4)
Maven(1)
MQ、JMS(2)
Netty、Mina、Grizzly(5)
NOSQL
Security(8)
Spring(19)
SQL(1)
Struts2(1)
Thrift(1)
Tomcat(18)
WebService、JAXWS、XFire
工具(5)
设计模式(5)
事务(3)
杂文(9)

随笔档案 (123)

2016年3月 (2)
2016年1月 (5)
2015年12月 (10)
2015年11月 (1)
2015年10月 (3)
2015年9月 (5)
2015年8月 (4)
2015年7月 (7)
2015年6月 (8)
2015年5月 (6)

```
/**
 * @author fs1194361820@163.com
 */
public class XYZProvider extends Provider{
    public XYZProvider(){
        super("XYZ", 1.0, "XYZ Security Provider v1.0");
        put("MessageDigest.XYZ", XYZMessageDigest.class.getName());
    }
}
```

已经默认配置了下列Provider：

```
security.provider.1=sun.security.provider.Sun
security.provider.2=sun.security.rsa.SunRsaSign
security.provider.3=sun.security.ec.SunEC
security.provider.4=com.sun.net.ssl.internal.ssl.Provider
security.provider.5=com.sun.crypto.provider.SunJCE
security.provider.6=sun.security.jgss.SunProvider
security.provider.7=com.sun.security.sasl.Provider
security.provider.8=org.jcp.xml.dsig.internal.dom.XMLDSigRI
security.provider.9=sun.security.smartcardio.SunPCSC
security.provider.10=sun.security.mscapi.SunMSCAPI
```

配置为：security.provider.11=com.fjn.security.XYZProvider 即可。

编码方式就更加简单了：Security.addProvider(new XYZProvider());

3) 消息摘要服务：MessageDigest

消息摘要服务其实就是使用hash算法将一段消息（可以是字符串、文件内容、html等）进行计算生成的一个byte[]。

2015年4月 (3)
2015年3月 (3)
2015年2月 (3)
2015年1月 (2)
2014年12月 (3)
2014年11月 (1)
2014年10月 (3)
2014年9月 (8)
2014年8月 (22)
2014年7月 (14)
2014年6月 (1)
2014年4月 (1)
2013年9月 (1)
2013年8月 (7)

最新评论

1. [Re: 让你脱离google不能访问的烦恼](#)
新的方案：使用SHADOWSOCKS，在手机、电脑上都可以使用的。可以上google、facebook、LinkIn等。下载地址以及使用方法参见：...


--螺丝钉

2. [Re: Tomcat源码解读：ClassLoader的设计](#)

Java中每个类都是由ClassLoader加载。一段代码想要能够正常的工作，这段代码中的每个类都要能够加载到才行。并不要求这段代码中的每个类的加载器是同一


常用加密算法MD5、SHA、SHA-1其实都是hash算法。

下面就给一个简单的MD5算法工具：

 [View Code](#)

Md5算法我并没有去实现，因为在JDK中已经内置了md5算法。上面的代码就是使用消息摘要服务，并使用md5算法，生成相应的摘要。

下面来一个自定义的MessageDigest：

 [View Code](#)

这个自定义的MessageDigest中备注的内容，其实就是MessageDigest的执行过程，所有的MessageDigest都要遵从这个过程的。

测试用例：

 [View Code](#)

在这个用例中，writeMessage()将一段字符串保存后并将生成的digest也保存。

readMessage()将消息读取后，使用MessageDigest.isEqual()方法进行比较，这样可以知道文件是否被人改动过。

个。如果没有指明类加载器，这个代码段中的类，.....

--螺丝钉

3. Re:Tomcat：利用Apache配置反向代理、负载均衡

@zeus2大大小小公司那么多，用法又不止那一种。难道每个公司用的都一样吗？...

--螺丝钉

4. Re:Tomcat：利用Apache配置反向代理、负载均衡

既然负载均衡了，硬件用F5 redware，软件用nginx，很少有人用什么apache的。

--zeus2

5. Re:Linux iptables 防火墙

RedHat 7 禁用防火墙的方法

```
[root@rhel7 ~]# systemctl stop  
firewalld.service[root@rhel7 ~]#  
systemctl disable f.....
```

--螺丝钉

6. Re:Thrift：快速开始

这不是菲哥么

--TobeOlder

7. Re:让你脱离google不能访问的烦恼

Google最近又不能上了，又找了个代理网站，共享一下：

而实际上利用私钥更新签名信息时，就是使用MessageDigest#update()方法的。

4) Key 相关的服务


Key包括公钥（PublicKey）、私钥（PrivateKey）两种。

4.1 KeyPairGenerator

这个服务用于生成PublicKey和PrivateKey。

```
initialize(int) : void  
initialize(int, SecureRandom) : void  
initialize(AlgorithmParameterSpec) : void  
initialize(AlgorithmParameterSpec, SecureRandom) : void
```

获取实例后，只需要根据上面4种initialize方法进行初始化后，就可以生成KeyPair了。

 View Code

第一次调用generateKeyPair()都会生成不同的KeyPair。KeyPairGenerator 每次生成的都是一个KeyPair。

4.2 KeyFactory

KeyFactory用于在Key与KeySpec之间转换，即可以根据key获取到KeySpec，也可以根据KeySpec获取Key。

--螺丝钉

8. Re:Thrift : 快速开始

支持一下!!!

--花儿笑弯了腰

9. Re:Java Se : Map 系列

@yyww谢谢了，当时没有明白你是在回答我在博客中问的问题。现在知道了，时隔这么长时间，不过也不算晚。...


--螺丝钉

10. Re:Java Se : 常见异常总结

你好，我是阿里巴巴架构师，觉得你blog写的不错，有兴趣看看阿里的机会吗？我邮箱：fulan.zjf@alibaba-inc.com

--significantfrank

阅读排行

 View Code

5) Cert相关的服务

从上一篇的例子中知道，用户使用的Public Key有可能被不法分子偷偷地篡改，这样用户就得不到应有的服务，也会受到不法分子的危害。如何保证public key不被篡改或者替换呢？认证服务就出现了。

5.1 CertificateFactory

用于生成Certificate或者CRL的。

1. JQuery插件：遮罩+数据加载中。。。
。（特点：遮你想遮，罩你想罩）(2159)
2. Spring源码阅读：IOC容器的设计与实现（二）——
ApplicationContext(1567)
3. Spring源码阅读系列总结(1479)
4. C++：主要知识点(1182)
5. JavaSE：你真的了解继承、重写、可见性吗？(1016)
6. 事务相关（将网络上的关于事务的文章大部分都整理了）(868)
7. Tomcat源码解读：Tomcat启动过程都干了啥(842)
8. Spring源码阅读：Spring AOP设计与实现（一）：动态代理(831)
9. Ajax跨域访问(807)
10. Spring源码阅读：Spring
WebApplicationContext初始化与消亡(801)

评论排行榜

1. 模拟JavaEE的Filter(10)
2. JavaSE：你真的了解继承、重写、可见性吗？(9)
3. Mina 快速入门(9)
4. Ajax跨域访问(9)
5. C++：主要知识点(7)

Certificate	generateCertificate (InputStream inStream) Generates a certificate object and initializes it
Collection <? extends Certificate >	generateCertificates (InputStream inStream) Returns a (possibly empty) collection view of the inStream.
CertPath	generateCertPath (InputStream inStream) Generates a CertPath object and initializes it with
CertPath	generateCertPath (InputStream inStream, String encoding) Generates a CertPath object and initializes it with
CertPath	generateCertPath (List <? extends Certificate > certificates) Generates a CertPath object and initializes it with
CRL	generateCRL (InputStream inStream) Generates a certificate revocation list (CRL) object from the input stream inStream.
Collection <? extends CRL >	generateCRLs (InputStream inStream) Returns a (possibly empty) collection view of the
Iterator < String >	getCertPathEncodings ()



```
FileInputStream fis = new FileInputStream(filename);
BufferedInputStream bis = new BufferedInputStream(fis);

CertificateFactory cf = CertificateFactory.getInstance("X.509");

while (bis.available() > 0) {
    Certificate cert = cf.generateCertificate(bis);
    System.out.println(cert.toString());
}
```



- 6. [Git 入门\(5\)](#)
- 7. [Java Annotation 学习\(5\)](#)
- 8. [Android APK 反编译\(5\)](#)
- 9. [Tomcat源码解读：我们发起的HTTP请求如何到达Servlet的\(4\)](#)
- 10. [Java Se：Java NIO（服务端）与 BIO（客户端）通信\(4\)](#)

推荐排行榜

- 1. [jQuery插件：遮罩+数据加载中。。。（特点：遮你想遮，罩你想罩）\(8\)](#)
- 2. [JavaSE：你真的了解继承、重写、可见性吗？\(7\)](#)
- 3. [Spring源码阅读系列总结\(5\)](#)
- 4. [把你的Project发布到GitHub上\(4\)](#)
- 5. [Tomcat源码解读：我们发起的HTTP请求如何到达Servlet的\(4\)](#)
- 6. [Java Annotation 学习\(3\)](#)
- 7. [Git 入门\(3\)](#)
- 8. [Linux iptables 防火墙\(3\)](#)
- 9. [Spring源码阅读：Spring事务管理的基础\(2\)](#)
- 10. [事务相关（将网络上的关于事务的文章大部分都整理了）\(2\)](#)

什么是CRL？

一个证书颁发机构需要证书吊销其颁发的证书——也许是虚假的,或者证书的用户已经使用证书从事非法行为。在这样的情况下,证书的有效期不足保护;证书必须立即失效。

下面的这个例子就是在验证完证书的有效性后，判断这个证书是否是一个吊销的证书。

 [View Code](#)

5.2 CertPathBuilder构建证书链CertPath

CertPath就是之前说的证书链。其实就是一个Certificate的有序列表。在列表的最后一个Cert是一个自签名的Cert。

5.3 CertPathValidator验证Cert链

CertPathValidator用于校验Cert。


6) KeyStore

一个KeyStore是一个key、cert的库，里面存储了PrivateKey, Aliases, Certs.

KeyStore将会有专门的说明。

7) Signature 签名

用私钥签名，用公钥验证：

 View Code

到此，JCA部分的engine class已经大体上有个了解了。接下来就是要学习如何应用它们了。

作者: 房继诺

出处: <http://www.cnblogs.com/f1194361820>

版权: 本文版权归作者和博客园共有

欢迎转载，**转载请需要注明博客出处**

技术交流QQ:**1194361820**，加好友请注明：**来自博客园**，不要说你是博客园，也可以扫描图像二维码直接加我。



分类: [J2SE](#), [Security](#)

标签: [Java Security](#), [Security](#), [PublicKey](#), [PrivateKey](#), [Certificate](#), [MessageDigest](#), [KeyStore](#), [KeyFactory](#), [Signature](#), [JCA](#)

好文要顶

关注我

收藏该文



救火队长

关注 - 8

粉丝 - 126

[+加关注](#)

0

推荐

0

反对

(请您对文章做出评价)

« [上一篇 : Java Security : 公钥私钥、数字签名、消息摘要是什么](#)

» [下一篇 : Java Security : keytool工具使用说明](#)

posted @ 2015-01-30 15:40 [救火队长](#) 阅读(465) 评论(0) [编辑](#) [收藏](#)

[刷新评论](#) [刷新页面](#) [返回顶部](#)



注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问](#) 网站首页。

【推荐】50万行VC++源码：大型组态工控、电力仿真CAD与GIS源码库

【推荐】融云即时通讯云—专注为 App 开发者提供IM云服务

【推荐】UCloud开年大礼，充5000返1000；买云主机送CDN，详情点击



最新IT新闻：

- 作为项目经理的7个经验教训总结
 - 这是全球首家全新设计的苹果店：配37寸巨屏背景墙
 - 首发VR游戏ADR1FT测评：沉迷宇宙与恶心到吐
 - 首席技术官 (CTO) 比普通程序员强在哪
 - Ubuntu 16.10看点
- » 更多新闻...



最新知识库文章：

- 为什么未来是全栈工程师的世界？
 - 程序bug导致了天大的损失，要枪毙程序猿吗？
 - 如何运维千台以上游戏云服务器
 - 架构漫谈（一）：什么是架构？
 - 架构的本质
- » 更多知识库文章...

