# 网址

https://www.cnblogs.com/lz2017/p/6917049.html

# 基本概念

JDK中带有部分加密算法的实现类，主要的是java.security和javax.crypto包下的类。

还可以使用Bouncy Castle（丰富JDK中加密算法的不足），jar包是：bcprov-jdk15on-1.57.jar和Commons Codec（简化JDK中加密的操作）jar包是：commons-codec-1.10.jar

## 加密算法

[数据加密](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%8A%A0%E5%AF%86/11048982)的基本过程就是对原来为明文的文件或数据按某种[算法](https://baike.baidu.com/item/%E7%AE%97%E6%B3%95/209025)进行处理，使其成为不可读的一段代码，通常称为“密文”，使其只能在输入相应的[密钥](https://baike.baidu.com/item/%E5%AF%86%E9%92%A5/101144)之后才能显示出本来内容，通过这样的途径来达到保护数据不被非法人窃取、阅读的目的。 该过程的逆过程为解密，即将该[编码](https://baike.baidu.com/item/%E7%BC%96%E7%A0%81/80092)信息转化为其原来数据的过程。

# Base64

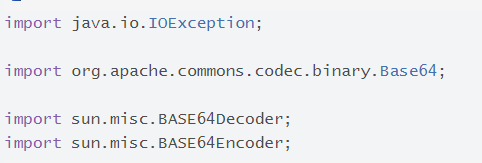
Base64用于网络中传输的数据进行编码，严格意义上属于编码的格式，有64个字符的对应的编码，Base64就是将内容按照该格式进行编码。可以对数据编码和解码，是可逆的，安全度较低，不过也可以作为最基础最简单的加密算法用于加密要求较弱的情况。

Base64可以使用JDk中自带的类实现，还可以使用Bouncy Castle（简称bc）或Commons Codec（简称cc）实现

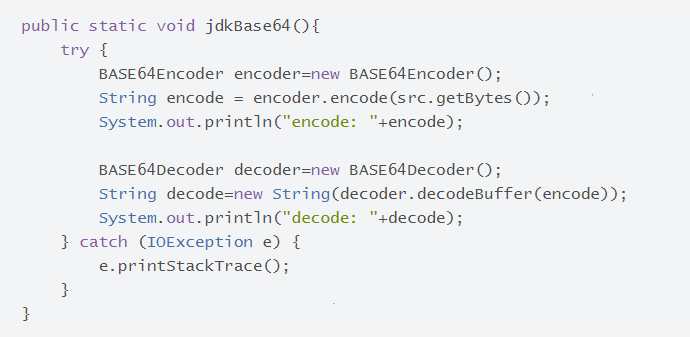
加密数据：

private static String src="Hello Base64";

导入的类：



JDk实现主要使用BASE64Encoder和BASE64Decoder类的方法（**注意**：在Eclipse中使用JDK的Base64可能会出现找不到的问题，是因为Base64Encoder并不属于JDK标准库范畴，但是又包含在了JDK中，需要我们手动导入\jre\lib目录下的rt.jar包即可）



bc实现主要是用Base64类的方法：



cc实现也是用Base64类，不过与bc的是不一样的，不同包中的类，只是名字一样



# 摘要算法

摘要算法主要分为MD，SHA和Hmac算法，摘要算法其实是用于效验数据完整性的，我们在下载某些文件时，会有MD5和SHA1值提供我们效验下载的文件是否完整，可以用于根据数据生成其唯一的摘要值，无法根据摘要值知道原数据，属于不可逆的。

## MD



加密数据：

private static String src="Hello MD";

导入的类：

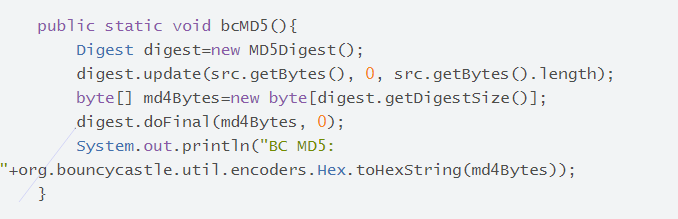


JDK有MD2和MD5的实现，使用的是MessageDigest类，而没有MD4的实现：



而bc这三种都有，使用的是Digest 类







cc和JDK是一样的，毕竟是对JDK加密的简化，直接使用DigestUtils中的方法，很简单，而且前两种方法还需要将MD值转换为十六进制，cc直接就帮我们转了：



## SHA

这里只是使用SHA-1，其他类型类似



加密数据：

private static String src="Hello SHA";

要导入的类：



JDK实现方式（同样是使用MessageDigest）



bc的实现方式（同样是Digest 类）

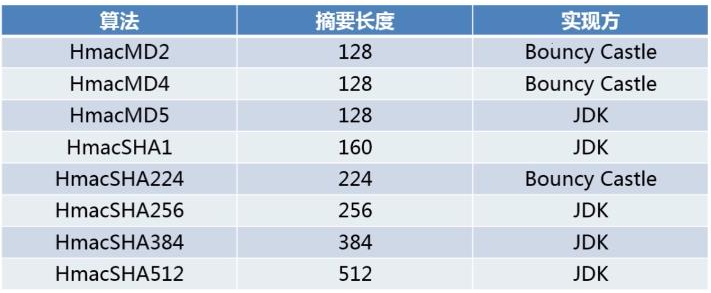


cc的实现方式



## Hmac

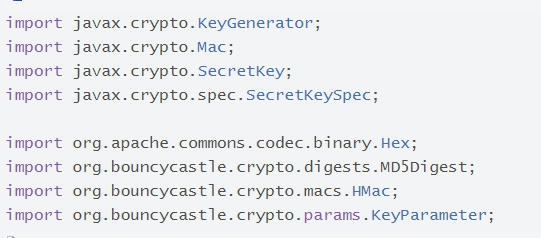
含有密钥的摘要算法，也有简称mac，密钥不同摘要也不同



要加密的数据：

private static String src="Hello HMAC";

要导入的类：



JDK的实现方式：



bc的实现方式：



# 对称加密算法

严格意义上的加密算法，分为对称和非对称加密算法，所谓对称是说发送方和接收方的密钥是一样的，而非对称我们后面再说。因为密钥一样所以安全性跟非对称比较来说就不太安全了。

对称加密算法主要分为:DES , 3DES(3重DES) , AES(想要替代DES)  , PBE(基于口令的对称算法)。

## DES



加密数据：

private static String src="Hello DES";

导入的类：



## AES



Java AES 加解密默认是支持128位的。

基本实现和DES类似，只不过在实现该算法的时候，设置密钥长度大于128会出现错误：Illegal key size or default parameters，这是因为美国的出口限制，Sun通过权限文件（local\_policy.jar、US\_export\_policy.jar）做了相应限制，Oracle在其官方网站上提供了无政策限制权限文件（Unlimited Strength Jurisdiction Policy Files），我们只需要将其部署在JRE环境中，就可以解决限制问题。

JDK8的无政策限制权限文件（http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html）

将下载的local\_policy.jar和US\_export\_policy.jar替换JDK的JRE环境中，或者是JRE环境中上述两个jar文件即可，非对称的ELGamal加密算法也有该问题，解决方法相同。

加密的数据：

private static String src="Hello AES";

导入的类：



### JDK的实现

默认是128位密钥长度



### Java AES256加密解密实现

代码如下：



### bc的实现

（同样使用了添加provider的做法）

