#### SHA1加密算法

SHA是一种数据加密算法，现在已成为公认的最安全的散列算法之一，并被广泛使用。该算法的思想是接收一段明文，然后以一种不可逆的方式将它转换成一段（通常更小）密文，也可以简单的理解为取一串输入码（称为预映射或信息），并把它们转化为长度较短、位数固定的输出序列即散列值（也称为信息摘要或信息认证代码）的过程。

散列函数值可以说是对明文的一种“指纹”或是“摘要”，所以散列值的数字签名可以视为对此明文的数字签名。

安全散列算法SHA（Secure Hash Algorithm，SHA）是美国国家标准技术研究所发布的。

#### SHA1算法原理

单向散列函数的安全性在于其产生散列值的操作过程具有较强的单向性。如果在输入序列中嵌入密码，那么任何人在不知道密码的情况下都不能产生正确的散列值，从而保证了其安全性。SHA将输入流按照每块512位（64字节）进行分块，并产生20个字节的被称为信息认证代码或信息摘要的输出。

该算法输入报文的长度不限，产生的输出是一个160位的报文摘要。输入是按512位的分组进行处理的。SHA-1是不可逆的、防冲突的，并具有良好的雪崩效应的。

通过散列算法可实现数字签名，数字签名的原理是将要传送的明文通过一种函数运算（Hash）转换成报文摘要（不同的明文对应不同的报文摘要），报文摘要加密后与明文一起传送给接受方，接受方将接受的明文产生新的报文摘要与发送方的发来报文摘要解密比较，比较结果一直表示明文未被改动，如果不一致表示明问已被篡改。

#### 算法Java实现

public class SHA1Utils {

public static String shaEncrypt(String str) throws Exception {  
  
 // 信息摘要算法（接收任意大小的数据，输出固定长度的哈希值）  
 MessageDigest sha = null;  
 try {  
 sha = MessageDigest.*getInstance*("SHA");  
 } catch (Exception e) {  
 System.*out*.println(e.toString());  
 e.printStackTrace();  
 return "";  
 }  
 // 输入UTF-8格式的字节数组  
 byte[] byteArray = str.getBytes("UTF-8");  
 byte[] mdBytes = sha.digest(byteArray);  
  
 StringBuffer hexValue = new StringBuffer();  
 for (int i = 0; i < mdBytes.length; i++) {  
 int val = ((int) mdBytes[i]) & 0xff;  
 if (val < 16) {  
 hexValue.append("0");  
 }  
 hexValue.append(Integer.*toHexString*(val));  
 }  
 return hexValue.toString();  
 }  
}

#### 实际应用中

在实际的应用中，我们可以直接引入org.apache.commons.codec.digest.DigestUtils，然后按照如下方法调用加密即可。

DigestUtils.*sha1Hex*(str);