漫谈Java加密技术（二）Java认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/644/2021_2022__E6_BC_AB_E8_B0_88Java_c104_644440.htm RSA 这种算法1978年就出现了，它是第一个既能用于数据加密也能用于数字签名的算法。它易于理解和操作，也很流行。算法的名字以发明者的名字命名：Ron Rivest， AdiShamir 和Leonard Adleman.这种加密算法的特点主要是密钥的变化，上文我们看到DES只有一个密钥。相当于只有一把钥匙，如果这把钥匙丢了，数据也就不安全了。RSA同时有两把钥匙，公钥与私钥。同时支持数字签名。数字签名的意义在于，对传输过来的数据进行校验。确保数据在传输工程中不被修改。 流程分析： 1、甲方构建密钥对儿，将公钥公布给乙方，将私钥保留。 2、甲方使用私钥加密数据，然后用私钥对加密后的数据签名，发送给乙方签名以及加密后的数据；乙方使用公钥、签名来验证待解密数据是否有效，如果有效使用公钥对数据解密。 3、乙方使用公钥加密数据，向甲方发送经过加密后的数据；甲方获得加密数据，通过私钥解密。 通过java代码实现如下： import java.security.Key. import java.security.KeyFactory. import java.security.KeyPair. import java.security.KeyPairGenerator. import java.security.PrivateKey. import java.security.PublicKey. import java.security.Signature. import java.security.interfaces.RSAPrivateKey. import java.security.interfaces.RSAPublicKey. import java.security.spec.PKCS8EncodedKeySpec. import java.security.spec.X509EncodedKeySpec. import java.util.HashMap.

import java.util.Map. import javax.crypto.Cipher. /** *//** * RSA安全编码组件 * * @version 1.0 * @since 1.0 */ public abstract class RSACoder extends Coder { public static final String KEY_ALGORITHM = "RSA". public static final String SIGNATURE_ALGORITHM = "MD5withRSA". private static final String PUBLIC_KEY = "RSAPublicKey". private static final String PRIVATE_KEY = "RSAPrivateKey". /** *//** * 用私钥对信息生成数字签名 * * @param data * 加密数据 * @param privateKey * 私钥 * * @return * @throws Exception */ public static String sign(byte[] data, String privateKey) throws Exception { // 解密由base64编码的私钥 byte[] keyBytes = decryptBASE64(privateKey). // 构造PKCS8EncodedKeySpec对象 PKCS8EncodedKeySpec pkcs8KeySpec = new PKCS8EncodedKeySpec(keyBytes). // KEY_ALGORITHM 指定的加密算法 KeyFactory keyFactory = KeyFactory.getInstance(KEY_ALGORITHM). // 取私钥匙对象 PrivateKey priKey = keyFactory.generatePrivate(pkcs8KeySpec). // 用私钥对信息生成数字签名 Signature signature = Signature.getInstance(SIGNATURE_ALGORITHM). signature.initSign(priKey). signature.0update(data). return encryptBASE64(signature.sign()). } /** *//** * 校验数字签名 * * @param data * 加密数据 * @param publicKey * 公钥 * @param sign * 数字签名 * * @return 校验成功返回true 失败返回false * @throws Exception * */ public static boolean verify(byte[] data, String publicKey, String sign) throws Exception { // 解密由base64编码的公钥 byte[] keyBytes = decryptBASE64(publicKey). // 构

造X509EncodedKeySpec对象 X509EncodedKeySpec keySpec = new X509EncodedKeySpec(keyBytes). // KEY_ALGORITHM 指定的加密算法 KeyFactory keyFactory = KeyFactory.getInstance(KEY_ALGORITHM). // 取公钥匙对象 PublicKey pubKey = keyFactory.generatePublic(keySpec). Signature signature = Signature.getInstance(SIGNATURE_ALGORITHM). signature.initVerify(pubKey). signature.0update(data). // 验证签名是否正常 return signature.verify(decryptBASE64(sign)). } 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com