**《密码学》课程设计实验报告**

实验序号：06　　　　　　　　　　实验项目名称：数字签名

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 学　　号 |  | 姓　　名 |  | 专业、班 | | 18信安 |
| 实验地点 | 新珈楼B310 | 指导教师 | 王后珍 | 时间 | | 2020.12.28 |
| 1. 实验目的及要求   实验目的：   1. 掌握数字签名的概念； 2. 掌握基于RSA密码、ElGamal密码和椭圆曲线密码的数字签名方法； 3. 了解基于RSA密码、ElGamal密码和椭圆曲线密码的数字签名的安全性； 4. 熟悉盲签名的原理，了解盲签名的应用。   实验要求：   1. 掌握RSA数字签名的实现方案； 2. 掌握ElGamal数字签名的实现方案； 3. 掌握SM2椭圆曲线数字签名的实现方案； 4. 了解数字签名实现中的相关优化算法。   二、实验设备（环境）及要求  Windows操作系统，高级语言开发环境  三、实验内容与步骤  1. 编程实现RSA数字签名方案  参数准备阶段：同实验07中的RSA密码算法  签名运算：  S＝Md mod n （8-5）  验证签名运算（判断式）：  M＝Se mod n （8-6）  实验1：采用实验06中的RSA密码算法的相关参数，对于M进行签名及验证。  思考1：RSA数字签名方案的几种攻击方法  思考2：基于RSA数字签名的盲签名方案的实现  2．编程实现ELGamal数字签名方案  复习数论的一个结论。对于素数*q*，如果是*q*的原根，则有： 取模(mod *q*)后各不相同。因此如果是*q*的原根，进一步有：   1. 对于任意整数m，当且仅当 2. 对于任意整数i，j，当且仅当   同ELGamal加密方案一样，ELGamal数字签名方案的基本元素是素数*p*和，其中是*p*的原根。用户A通过如下步骤产生公钥/私钥对：   1. 生成随机整数，使得。 2. 计算。   A的私钥是；A的公钥是。  **例：**设p=19，m=14,构造一个ELGamal数字签名方案，并用它对m签名。  对于p=19，原根有{2,3,10,13,14,15}，任选其中之一作为模 19的本原元（生成元），如选择  步骤1：**密钥生成**  用户随机地选择一个整数x作为自己的秘密的解密钥，1<x<p-1，计算y ≡αx mod p，取y 为自己的公开的加密钥。例如选择，，即私钥为16，公钥为4.  步骤2：**签名过程**  将明文消息M（0≤M≤p-1)加密成密文的过程如下：   1. 随机地选取一个整数k，k与p-1互素且1≤k≤p-1。例如随机选择 2. 计算   ③ 取（r ，s）=（3,4）作为m=14的签名。  步骤3：**验证过程**  对签名（r ，s）验证的过程如下：   1. 计算 2. 计算   由于，所以签名是合法的。  实验2：采用实验07中的ELGamal密码算法的相关参数，对于M进行签名及验证。  实验3：任意选作教材p254表8-1中的数字签名的变形算法，对于M进行签名及验证。  实验4：设，根据签名算法的一般形式，以及对应的验证算法的一般形式，自己尝试设计新的基于离散对数的数字签名方案，并对于M进行签名及验证。（选作）  3. 编程实现SM2椭圆曲线数字签名方案（选作）  思考1：椭圆曲线加密、签名的快速实现；  提示1：模参数、曲线参数的选取优化；  提示2：点加和倍点运算的快速实现；  思考2：k=15时，kP运算次数  反复平方乘 31P=[11111]2P=2(2（2（2P+P）+P）+P)+P，共4次加法、4次倍加  改进编码31P=（25-1）P=2(2(2(2(2P))))-P，需要5次倍加，1次加法（减法）  四、实验结果与数据处理  ***RSA签名代码整体说明***  相关文件(rsa.py)  **由于在前面的RSA加解密(实验报告5)中已经对RSA的相关接口进行了介绍，这里我们不再重复。**  这里主要给出签名和验签的流程图  观察其签名和验签的流程，我们可以发现其实就是**私钥解密，公钥加密**    ***RSA***签名    ***RSA签名核心代码说明***  **由于在实验报告5中已经对RSA的相关代码进行了分析，这里主要给出签名和验签的代码**。  在对某个二进制文件进行签名之前，我们首先需要计算其数据摘要，这里我们采用md5哈希算法，其对应的代码如下    ***md5***哈希算法    在RSA的签名中，其实就是用**私钥进行解密**的过程，其对应的代码如下    ***RSA***签名  其中解密对应的代码如下，***pow***函数提供了快速幂的功能    ***RSA***解密  在RSA验签的过程中，其实就是用**公钥进行加密**的过程，其对应的代码如下    ***RSA***验签    其中加密对应的算法如下    ***RSA***加密  ***RSA签名实验结果演示***  首先是计算待签名文件的md5值    ***md5***结果  对得到的md5值进行签名，可以得到如下的结果    ***RSA***签名结果    对签名进行验签，可以发现成功验证签名    验证签名  ***ECC签名代码整体说明***  相关文件(ecc.py)  **由于在前面的ECC加解密(实验报告5)中已经对ECC的相关接口进行了介绍，这里我们不再重复。**  这里主要给出签名和验签的主要流程  签名的主要流程如下   1. 选择一个随机数， 2. 计算点，并记 3. 利用保密的解密钥计算 4. 以作为消息的签名，并以的形式传输或者存储   验签的主要流程如下   1. 计算 2. 利用公开的加密钥计算 3. 如果，则是用户对的签名   ***ECC签名核心代码说明***  **由于在实验报告5中已经对ECC的相关代码进行了分析，这里主要给出签名和验签的代码**。  进行签名的代码主要如下，可以发现和我们上面提到的签名步骤是一一对应的    ***ECC***签名  进行验签的代码如下，和上述我们提到的验签步骤也是一一对应的    ***ECC***验签  ***ECC签名实验结果演示***  同RSA签名一样，我们首先需要计算文件对应的数据摘要，这里我们使用md5哈希算法    ***md5***数据摘要  接着对数据摘要进行签名，得到如下的签名结果    ***ECC***签名结果  对得到的签名进行验签，可以发现成功验证    ***ECC***验签  五、分析与讨论   1. 与ECC的签名相比，RSA的签名和验签显得更为优雅，即解密对应签名，加密对应验签 2. 和RSA签名相比，ECC引入了随机数k，使得即使对同一个数据进行签名，产生的结果也可能不一样，这在一定程度上增加了ECC签名的安全性 3. 与传统的分组密码相比，使用公钥密码实现数字签名非常方便且更加安全 | | | | | | |
| 六、教师评语  签名：  日期： | | | | | 成绩 | |

|  |
| --- |
|  |