项目一实验报告

1. 实验目的

设计并实现⼀个具备基本功能的KMS、CA和SSL/TLS之外的⽹络安全协议。

并模拟该协议或者应⽤ 场景的执⾏。

1. 选取加密与签名算法

加密算法：AES算法（因为对称加密没有公钥，所以证书中的公钥为签名算法的公钥）

签名方案：DSA签名方案，哈希算法选择SHA256

1. KMS模块

**1.设计目标**

本项目KMS模块的设计目标是：

·提供基础的密钥生命周期管理功能（生成、更新、撤销、备份与恢复）；

·支持用户之间的密钥访问控制；

·使用 Python 脚本通过命令行操作进行模拟调用；

·为后续的文件加密传输协议或身份认证协议提供密钥服务接口。

**2.系统架构与实现概览**

| **项目** | **工具或库** | **说明** |
| --- | --- | --- |
| 开发语言 | Python 3 | 易于开发与调试 |
| 加密算法库 | cryptography | 提供 AES 对称密钥生成功能 |
| 数据存储 | 本地 JSON 文件 | 模拟数据库存储用户密钥与访问权限 |
| 命令行交互 | argparse 库 | 用于封装 CLI 接口 |

**3.功能模块与代码解析**

(1)密钥生成与注册

功能说明：

为新用户生成一个 256 位的对称密钥（用于 AES 加密），并以 Base64 编码形式存储在本地数据库（JSON 文件）中。



·AESGCM.generate\_key(bit\_length=256) 创建一个 256 位的对称密钥；

·使用 Base64 编码密钥，便于以字符串形式安全存储；

·若用户名重复则阻止重复注册。

(2)密钥查询与访问控制

功能说明：

用户可查询自身密钥；他人需经过授权才能访问该用户密钥，实现基本访问控制机制。



·用户可通过 --as 参数模拟以他人身份获取密钥；

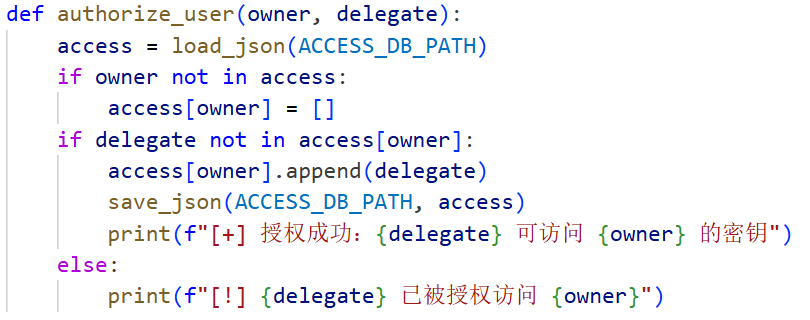
·访问权限通过 access\_control.json 控制；

·使用 access[username] = [delegates] 存储授权访问者列表。

(3)授权机制

功能说明：

用户可授权其他用户访问其密钥。



·授权信息持久化保存；

·同一用户可授权多个访问者；

·无需对称地授权（单向有效）。

(4)密钥更新与撤销

功能说明：

支持密钥替换（更新）和彻底删除（撤销），实现密钥生命周期管理。



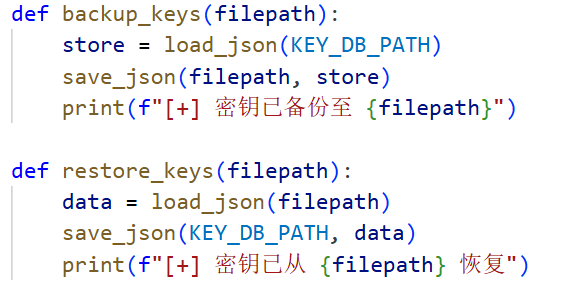
·更新：生成新密钥替换旧密钥；

·撤销：从系统中删除用户密钥（无法恢复）。

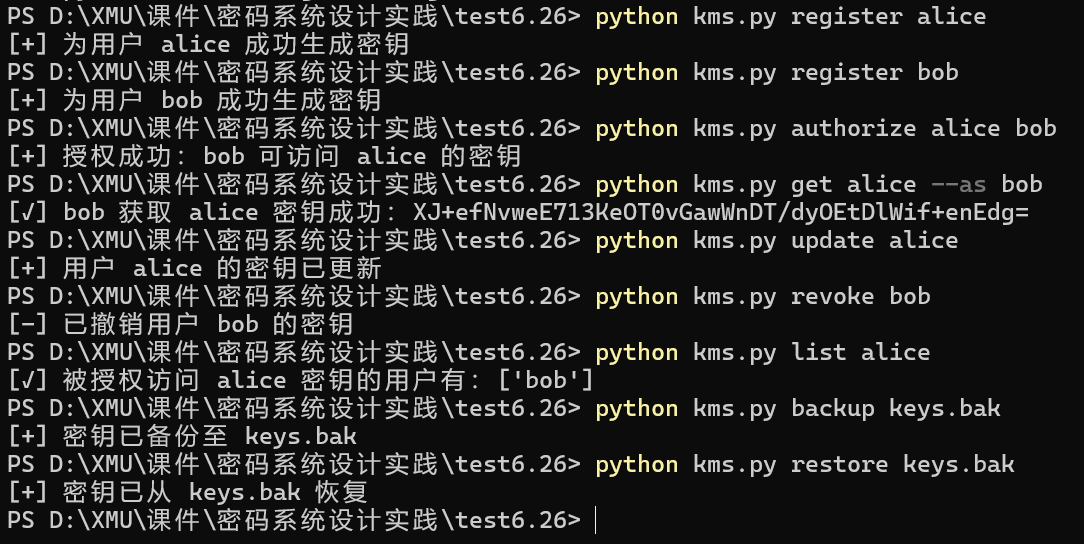
(5)密钥备份与恢复

功能说明：

为防止数据丢失，支持将密钥数据库导出到文件或从文件中恢复。

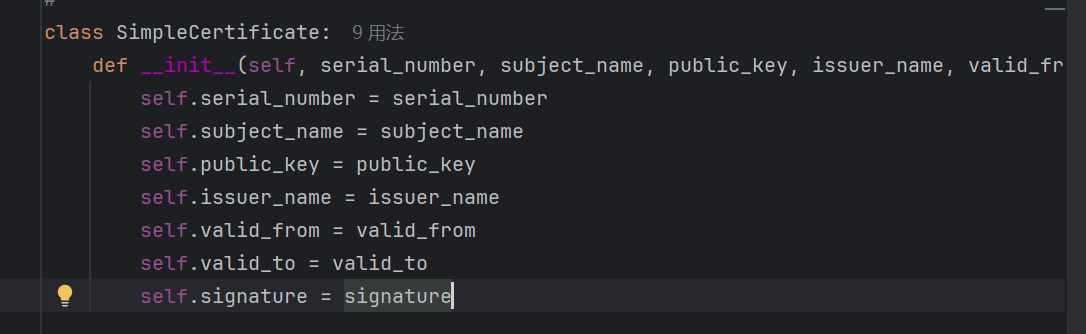


**4.命令行交互示例**



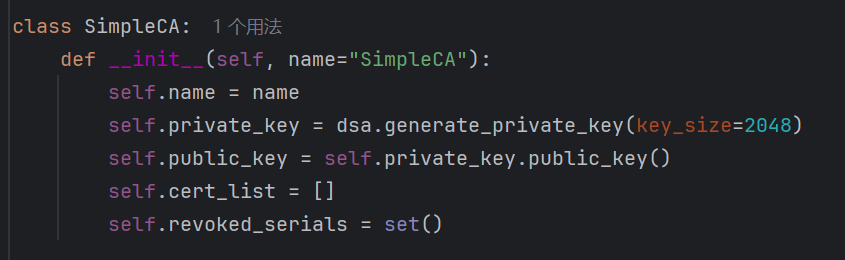
1. CA模块
2. 证书结构

证书为自定义结构，仅包含证书序列号、持有者名称、持有者公钥、签发者名称、生效时间、过期时间、CA对信息的签名。



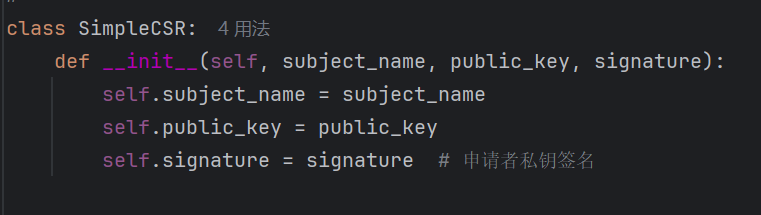
1. CA结构

包含名称、私钥和公钥、证书列表、被吊销的证书列表。



1. CSR结构

包含申请方姓名，公钥以及签名



1. 创建CSR与验证CSR

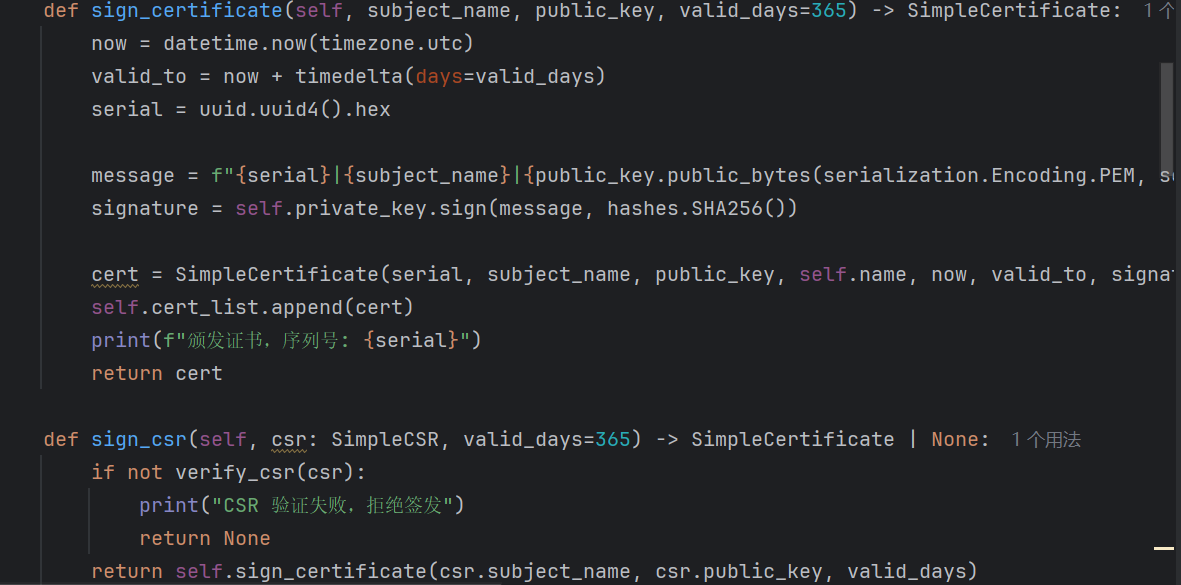
申请方通过自己的签名算法公私钥进行签名，CA进行验证，从而确认申请方确实拥有这个公钥。



1. 创建证书部分

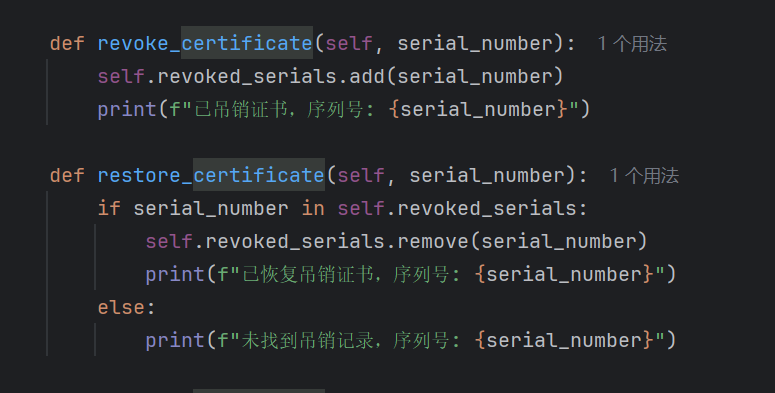
首先收到CSR申请之后，验证CSR的正确性，验证通过调用sign\_certificate

生成证书，默认有效时间为1年



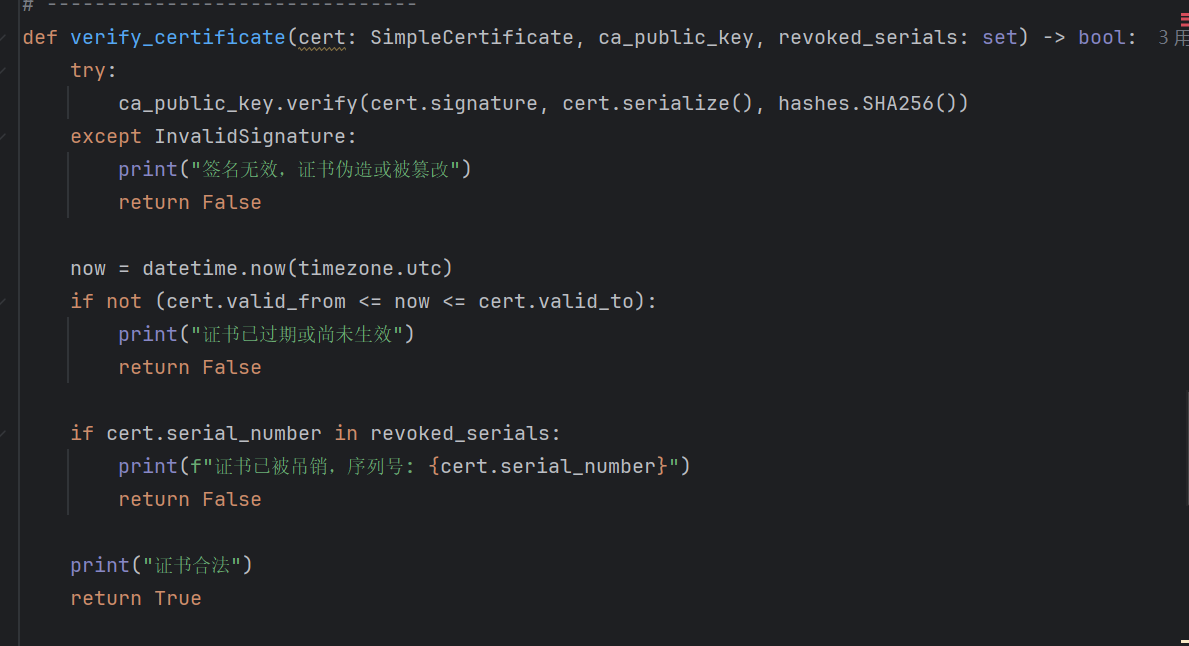
1. 证书的吊销与恢复部分

吊销将证书移入到revoked\_serials中，恢复将其移出。



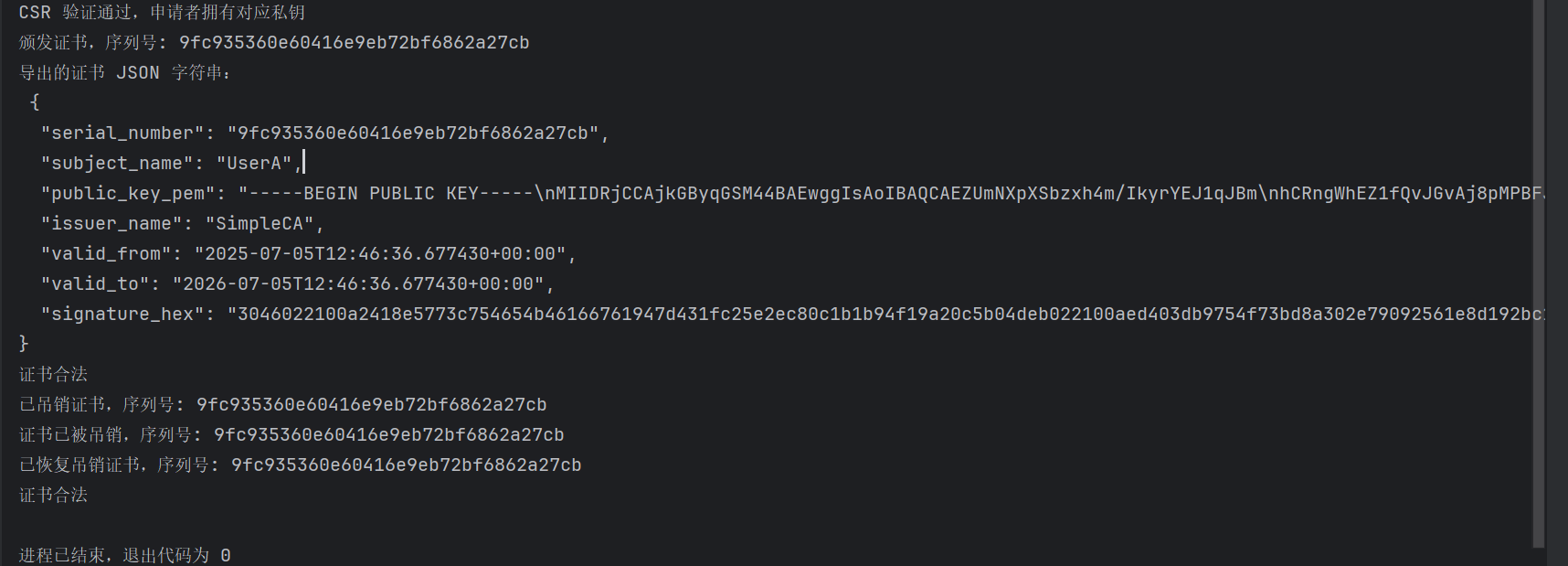
1. 验证证书部分

首先进行解密，然后判断有效期，以及是否在吊销列表



1. 测试与结果





1. 文件加密传输模拟模块

1、首先需要从KMS和CA模块中导入所需功能。

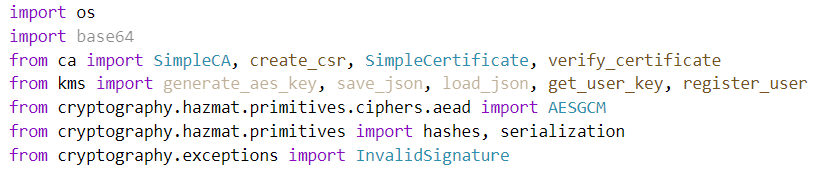


图 1 导入功能模块

2、当一个用户需要发送或接收文件时，首先都需要在KMS中进行注册，生成之后在文件加密传输时使用的密钥，并存储在KMS中；对于签名模块，在生成自己的私钥后，还需要向CA申请证书，证书包含身份信息以及用户的公钥，可以证明公钥确实是属于该用户的。申请完成后将私钥和证书单独保存成文件，便于之后调用。



图 2 用户注册及初始化

3、文件传输前需要经过加密和签名两个过程。首先从KMS中获取接收方注册时生成的密钥，因为本项目基于对称加密，使用的是AES加密。保存密文以及进行签名时在加密结果前拼接一个nonce随机数，效果类似于加盐salt，保证加密结果唯一。本项目使用DSA签名方案，哈希算法使用SHA256.同样将密文和签名单独保存成文件。



图 3 文件加密及签名

4、发送方完成文件的加密传输后，接收方不能马上对文件进行解密查看，因为文件可能是伪造的或存在损坏。所以接收方需要先验证发送方的证书，验证通过则证明发送方的公钥有效，再使用公钥验证文件签名，签名有效后才会从KMS中获取AES密钥，需要注意解密时需要将前12位nonce去掉，因为加密时并没有拼接上nonce。



图 4 文件接收

5、设置发送方Alice和接收方Bob两个用户，首先为两个用户申请证书并注册KMS。

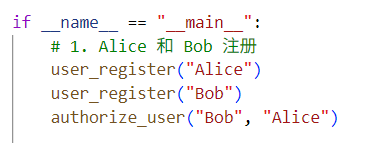


图 5 创建用户

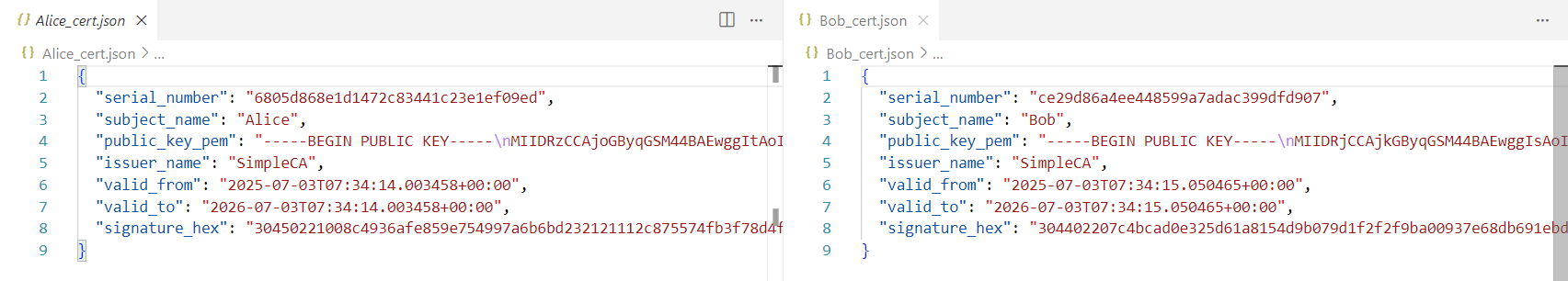


图 6 证书

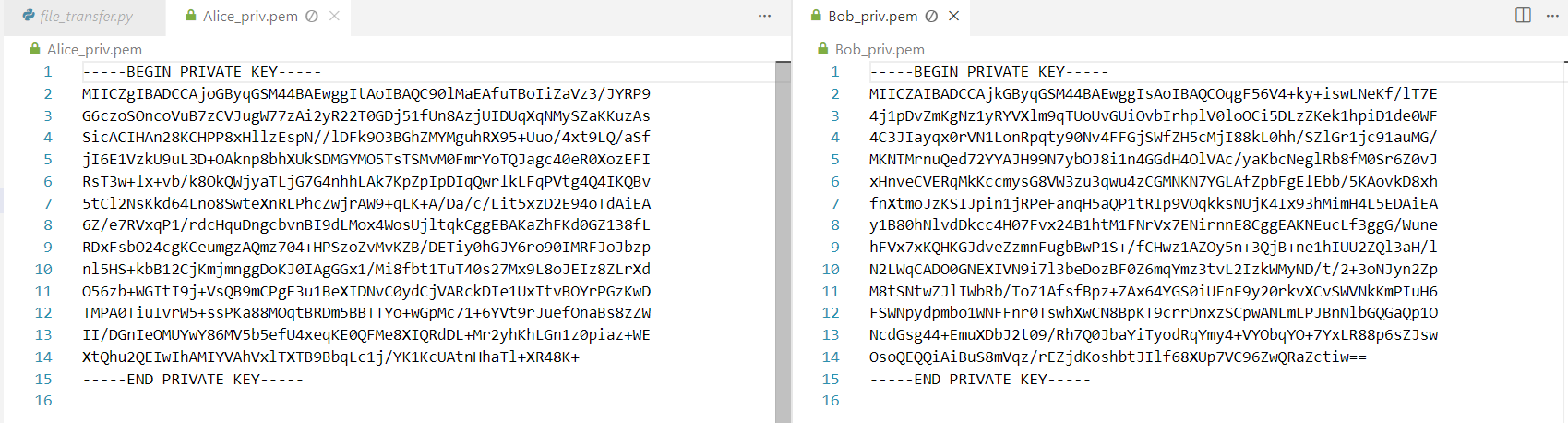


图 7 私钥

6、Alice将文件加密签名后发送给Bob，Bob验证证书和签名后解密得到文件。

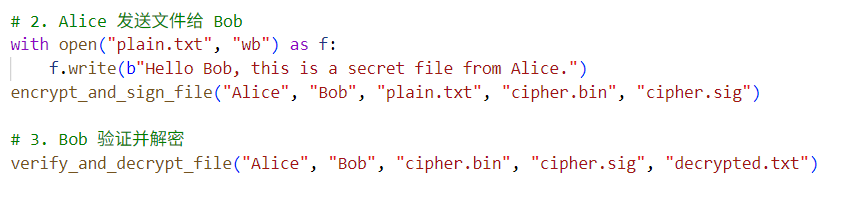


图 8 文件发送和接收

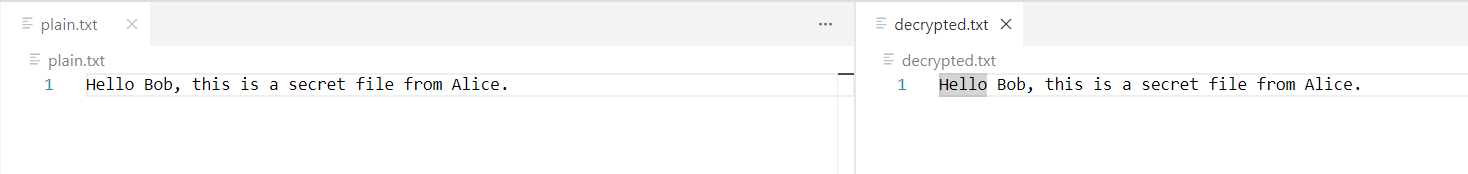


图 9 文件传输成功

1. 如果Alice是发送方则文件加密使用的是Bob注册的密钥，否则使用的是Alice注册的密钥，总的来说是接收方注册的密钥，要授权发送方访问接收方的密钥。需要注意这个密钥和证书中的公钥不同，证书中的公钥是DSA签名方案的公钥。



图 10 Alice发送给Bob



图 11 Bob发送给Alice

1. 小组分工

徐世杰 CA模块实现以及报告PPT对应部分

荀渤凯 KMS模块以及报告PPT对应部分

庞轶轲 文件加密传输模拟模块以及报告PPT对应部分