北京电子科技学院（BESTI）

**实 验 报 告**

课程：信息安全系统设计 班级：2214

姓名：徐鹿鸣 学号：20221414

成绩： 指导教师：娄嘉鹏 实验日期：2024/10/20

实验密级：无 预习程度：全预习

实验时间：1,2节 仪器组次：无 必修/选修：必修

实验序号：实验一

实验名称： 嵌入式开发基础

实验目的与要求：掌握Linux系统使用与开发方法。

实验仪器：

|  |  |  |
| --- | --- | --- |
| 名称 | 型号 | 数量 |
| 计算机 |  | 1 |
|  |  |  |

实验内容、步骤与体会（附纸）：

**1. 实验内容**

**Linux命令，OpenSSL（GmSSL）命令与开发。**

**2. 实验步骤**

**2.1步骤1 在Ubuntu中实践openssl与gmssl命令**

**2.1.1 操作内容**

**进入娄老师的《密码系统设计》思维导图课件，找到openssl和gmssl的相关命令。提前在Ubuntu中按照openssl、gmssl、useopenssl和usegmssl建立文件夹，分别对应实验1-1的四个部分。在对应文件夹中直接复制娄老师的参考命令进行实验。部分命令的内容需要修改，比如随机质数检验、一些以名字缩写命名的文件。部分命令会报错，此时将报错的命令和信息告诉AI，寻求解答。**

**2.1.2操作结果**

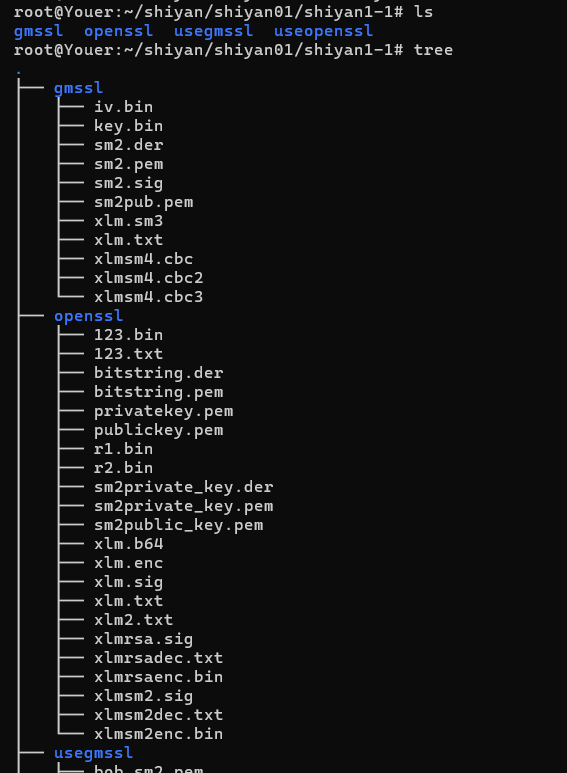


图 1 项目结构和具体文件展示

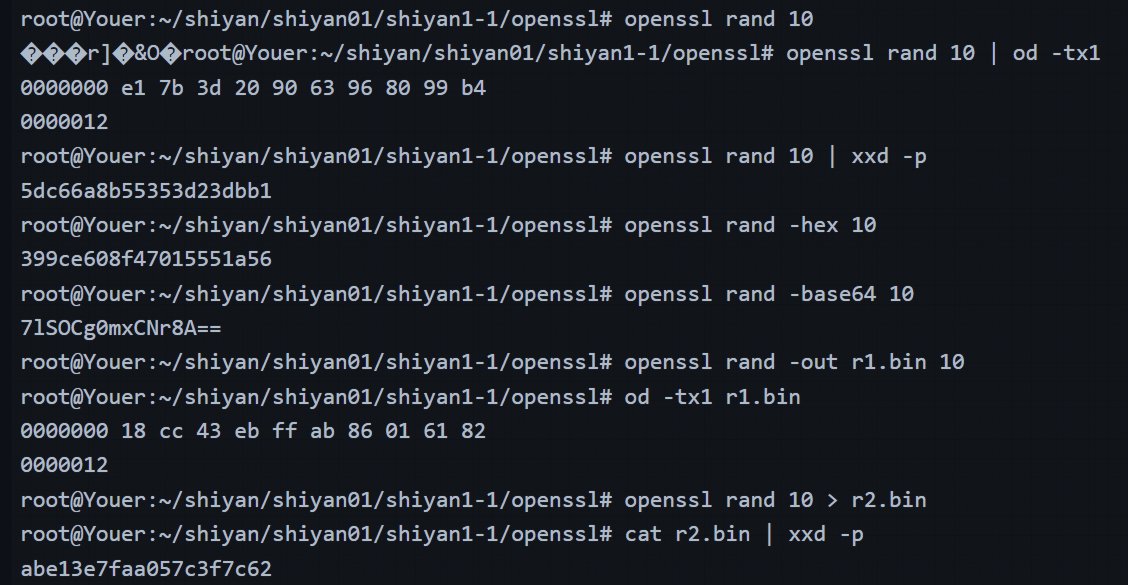


图 2 过程展示（openssl的rand命令）

**2.2步骤2 与陆宇航合作完成基于openssl命令或gmssl命令的带签名的数字信封协议**

**2.2.2 操作内容**

**这次实验我们统一采取Alice发送数字信封，Bob接受并验证的形式。在基于openssl命令实现时，我当Alice；在基于gmssl实现时，我当Bob。实验过程按照《实验要求(stu)》文件的要求进行，分别在useopenssl和usegmssl文件夹中执行操作。实验命令一般从娄老师的思维导图课件中获取，但也结合之前实践的经验对部分命令进行修改，特别是sm2签名和验证的命令。实验中的公钥文件、IV文件和数字信封文件都通过QQ邮箱相互传输。**

**2.2.2 操作结果**

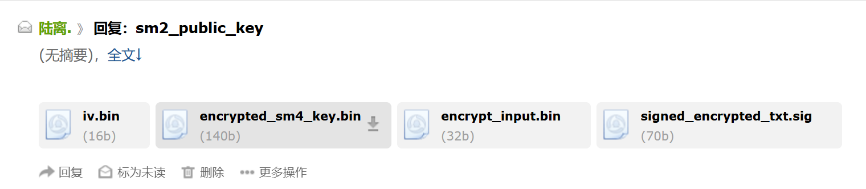
****

图 3 通过QQ邮箱传递公钥等文件

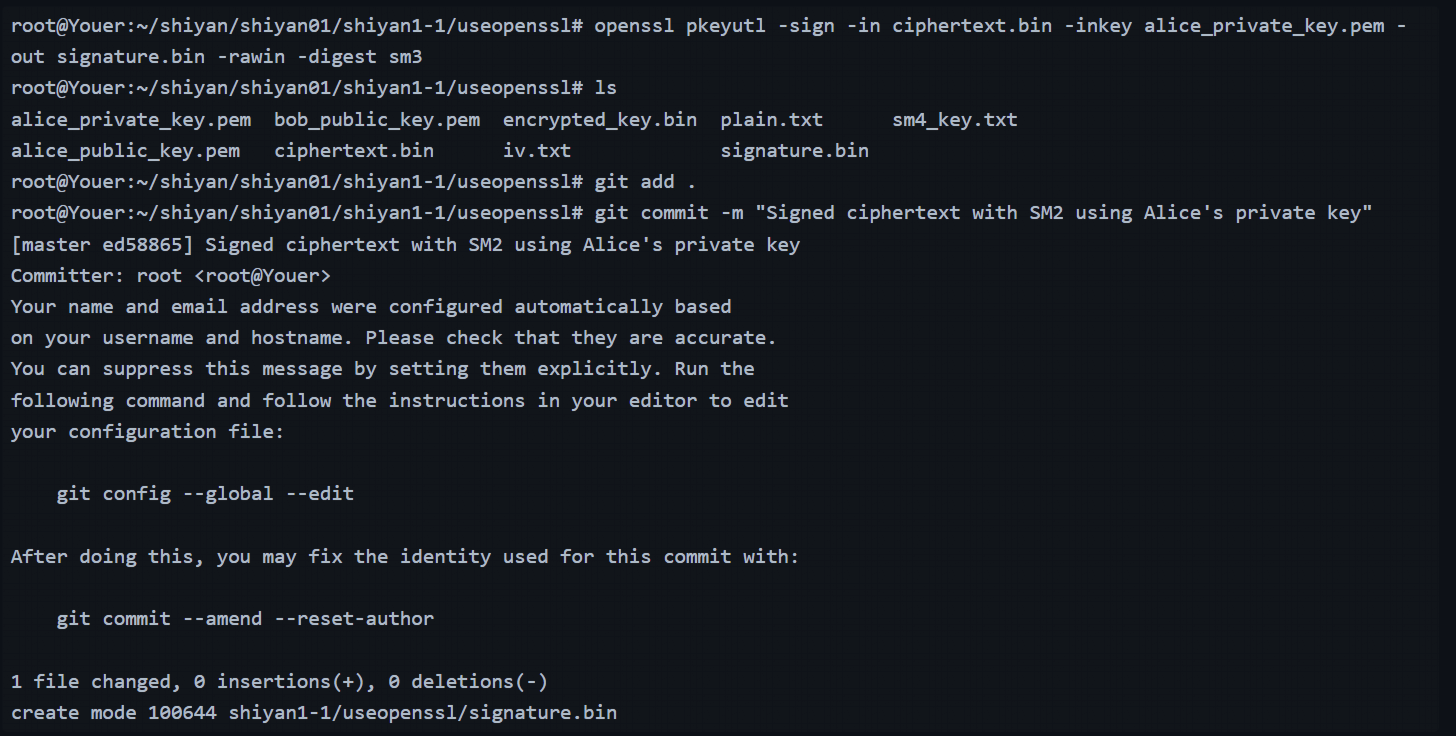
****

图 4 实验过程（这一步是Alice：Sm2Sign（SKa，C）= S1与相关的git提交）

**2.3 步骤3：使用OpenSSL库编程实现调用SM2（加密解密，签名验签），SM3（摘要计算，HMAC 计算），SM4（加密解密）算法**

**2.3.1 操作内容**

**首先建立task01文件夹，在task01文件夹中建立test\_sm2、test\_sm3、test\_sm4文件夹。在对应文件夹中用代码实现相关算法。**

**实现过程中，首先向AI提问“请为我在Ubuntu中调用openssl库实现XXX算法的c语言程序”，使用nano编辑器写入得到的程序，通过AI提示的编译命令进行编译。如果报错，就将报错信息返回AI，根据报错信息修改提问，让AI重新回答。直到编译成功，就进行功能验证。如果失败，将代码和报错信息发给AI，让AI继续回答。用rm删除旧的代码文件，用nano命令创建新的代码文件，重新编译。循环这个过程直到成功。**

**同时，在整个过程中，如果对一个AI重复询问太多次，就要放弃该对话，择优一份代码，调整自己的提问，必要时更换AI，重新开启对话。在新的提问中要学习总结经验，让AI避免犯一些常见的问题，比如通过指定库的版本避免调用旧的API、告诉AI哪些是错误的头文件。**

**在刚开始实验时，我的sm2密钥是通过openssl命令生成的，后续将其换成代码生成；我的SM4密钥和IV文件都是直接输入明文的，后续改为用gmssl命令生成密钥文件和IV文件。**

**在实验过程中，我渐渐明白openssl本身不太支持sm2等算法，都是通过EVP接口来实现相关操作。所以在询问AI时要指明使用EVP接口，同时删除其提供的错误头文件（比如”openssl/sm2.h”）。**

**2.3.2 操作结果**

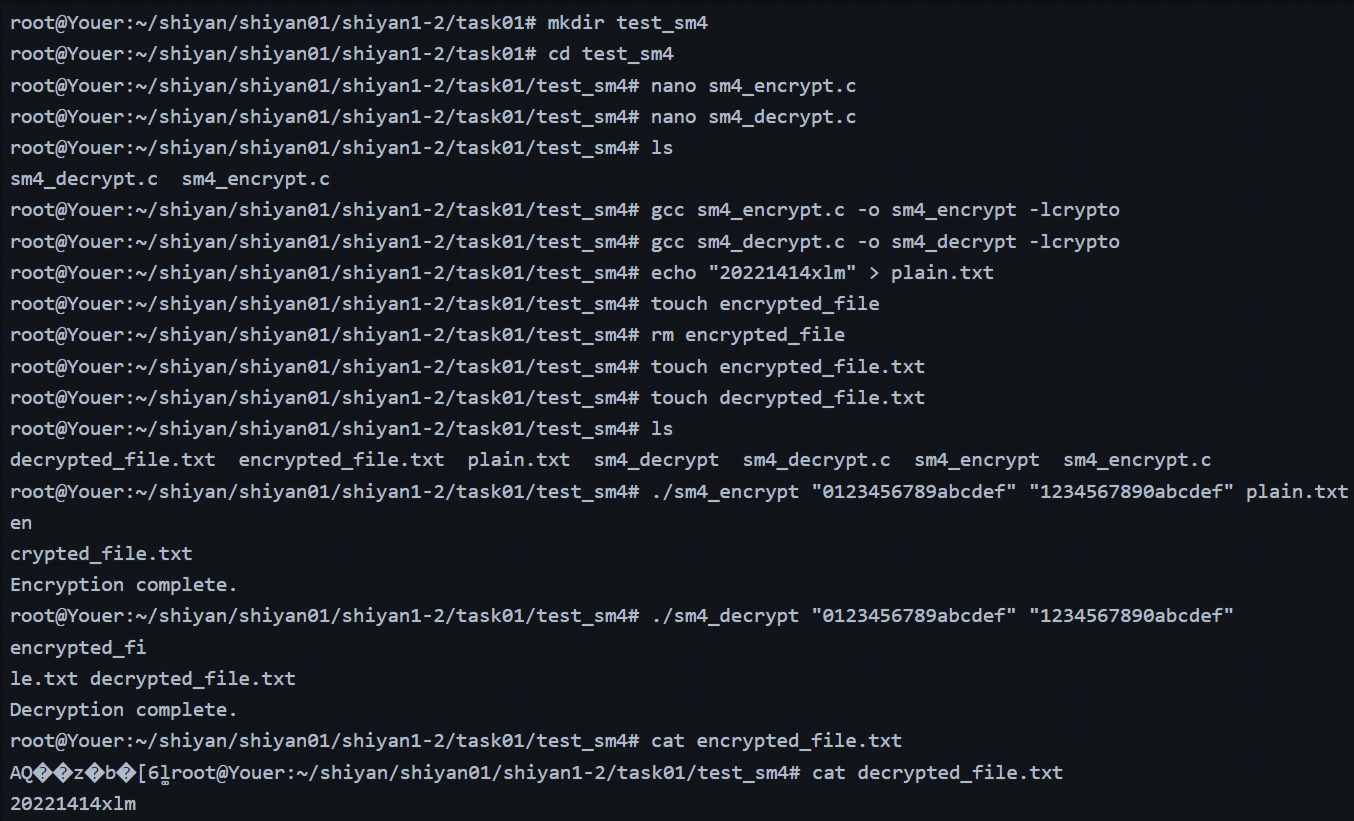
****

图 5 过程展示（图中是SM4加解密代码创建和验证过程）

****

图 6 AI对话截图（部分对话标题）

**2.4 步骤4：使用GmSSL库编程实现调用SM2（加密解密，签名验签），SM3（摘要计算，HMAC 计算），SM4（加密解密）算法**

**2.4.1 操作内容**

**建立task02文件夹，在task02的下一层保持与上一步一样的文件架构，但在每个文件夹中建立src文件夹存储c文件，建立bin文件夹存储可执行文件，建立test文件夹存储一些用来测试的文件。**

**实现过程中，和上一步骤一样询问AI。同样需要注意询问技巧，不要做无用功。**

**在实验过程中，我第一次没能用C语言直接实现SM3的HMAC功能，所以用代码来生成gmssl命令来实现相关效果。后续用python调用Gmssl库实现了SM3的HMAC算法。**

**2.4.2 操作结果**

****

图 7 询问AI截图（这里是尝试生成一个SM4加密程序）

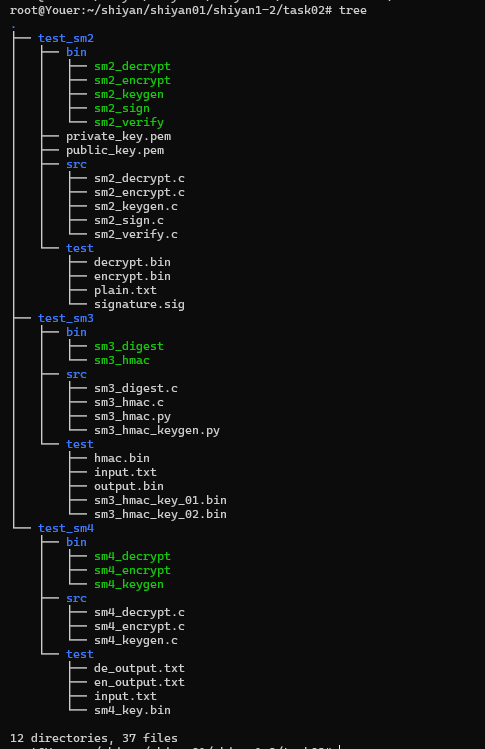
****

图 8 文件架构

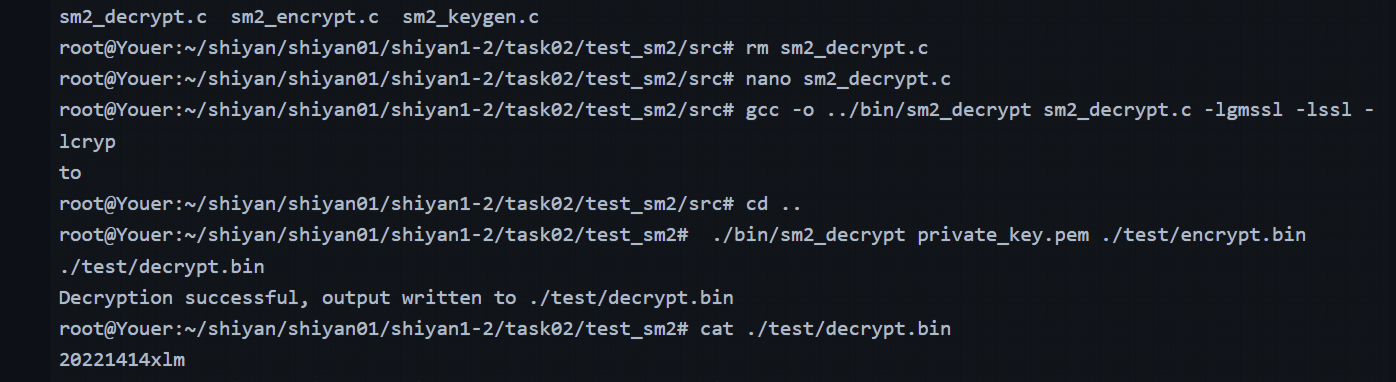
****

图 9 实现过程（这里是SM2的解密）

**2.5 步骤5：使用GmSSL库编程的代码实现带签名的数字信封**

**2.5.1 操作内容**

**我与陆宇航合作完成这一步骤。在这一步中，我是Alice，并负责生成和发送带签名的数字信封。**

**首先是建立task04文件夹，将task02文件夹中已经生成的可执行文件都拷贝过来，放在bin文件夹中。**

**按照实验要求生成公私钥对、明文、sm4密钥，交换公钥，用sm4密钥加密明文，用对方的公钥加密sm4密钥，用自己的私钥对密文签名。将密文、sm4密钥加密结果、签名合在一起发送给bob（陆宇航）。 至此，Alice的工作完成。**

**2.5.2 操作结果**

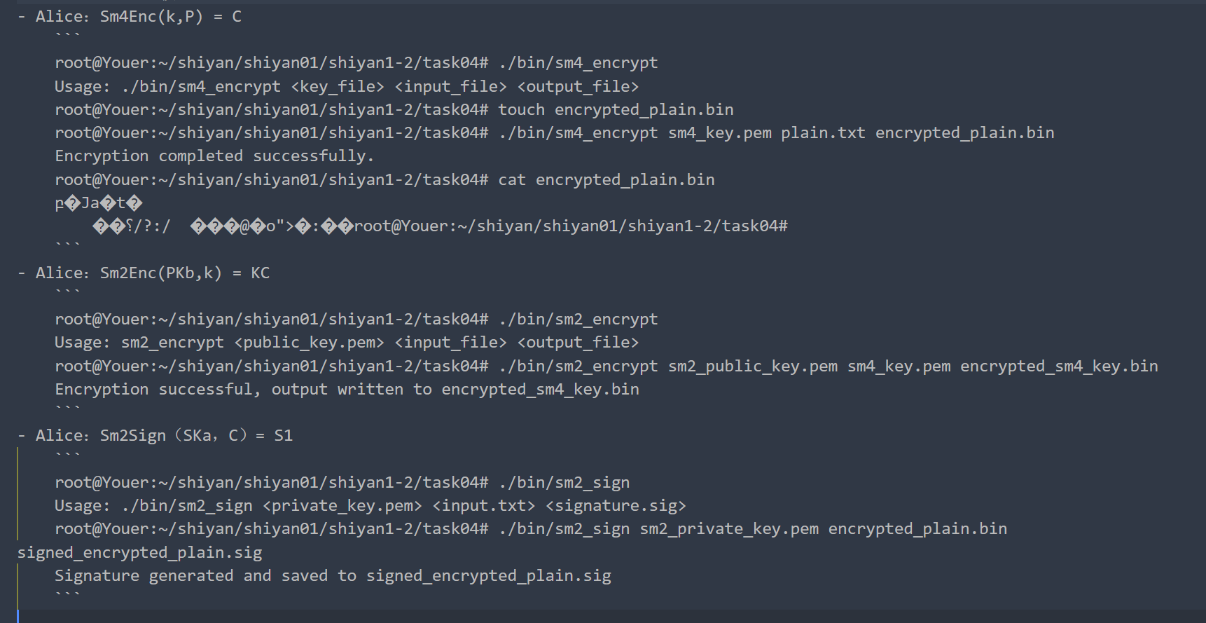
****

图 10 部分过程展示

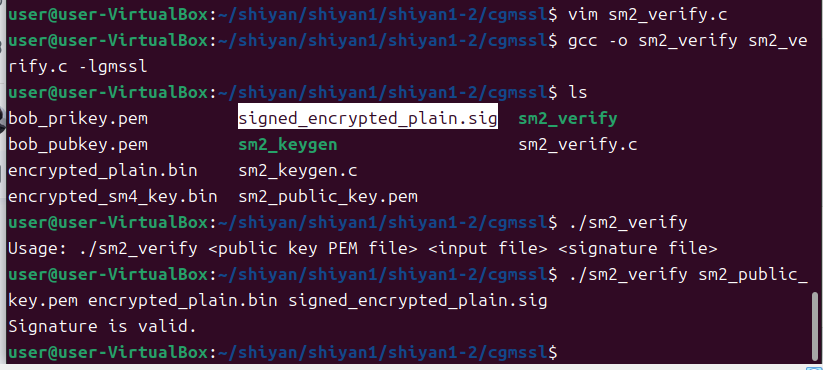
****

图 11 Bob（陆宇航）的签名验证结果

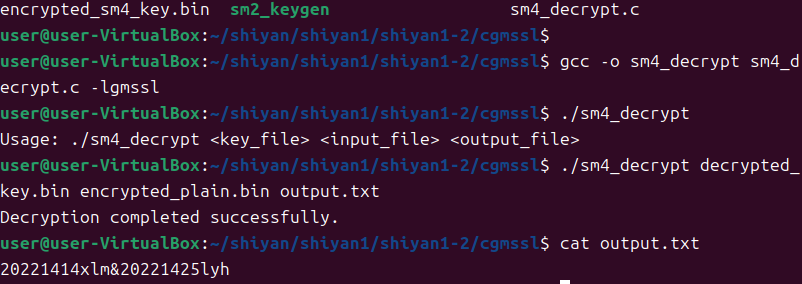
****

图 12 Bob（陆宇航）的解密结果

**2.6 步骤6：使用OpenSSL库编程的代码实现带签名的数字信封**

**2.6.1 操作内容**

**我和陆宇航合作完成这一步骤。在这里，我是Bob，负责接受数字信封并验证其是否正确。**

**首先建立task03文件夹，将task01文件夹中已经生成的可执行文件都拷贝过来，放在bin文件夹中。**

**按照实验要求生成公私钥对，交换公钥；获得Alice传过来的密文、被加密的密钥和签名结果；用Alice的公钥验证签名是否正确；用自己的私钥解密加密的密钥得到sm4密钥；用sm4密钥解密密文得到明文。查看明文是否与Alice原先的明文一致。经实验，签名验证正确，明文验证正确。**

**2.6.2 操作结果**

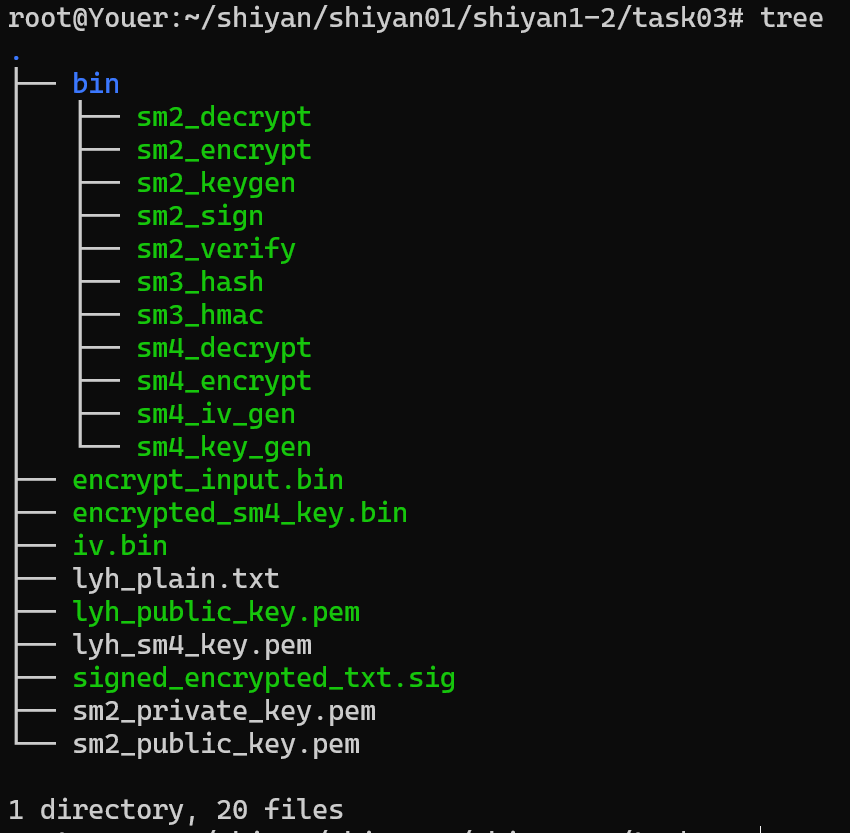
****

图 13 文件架构

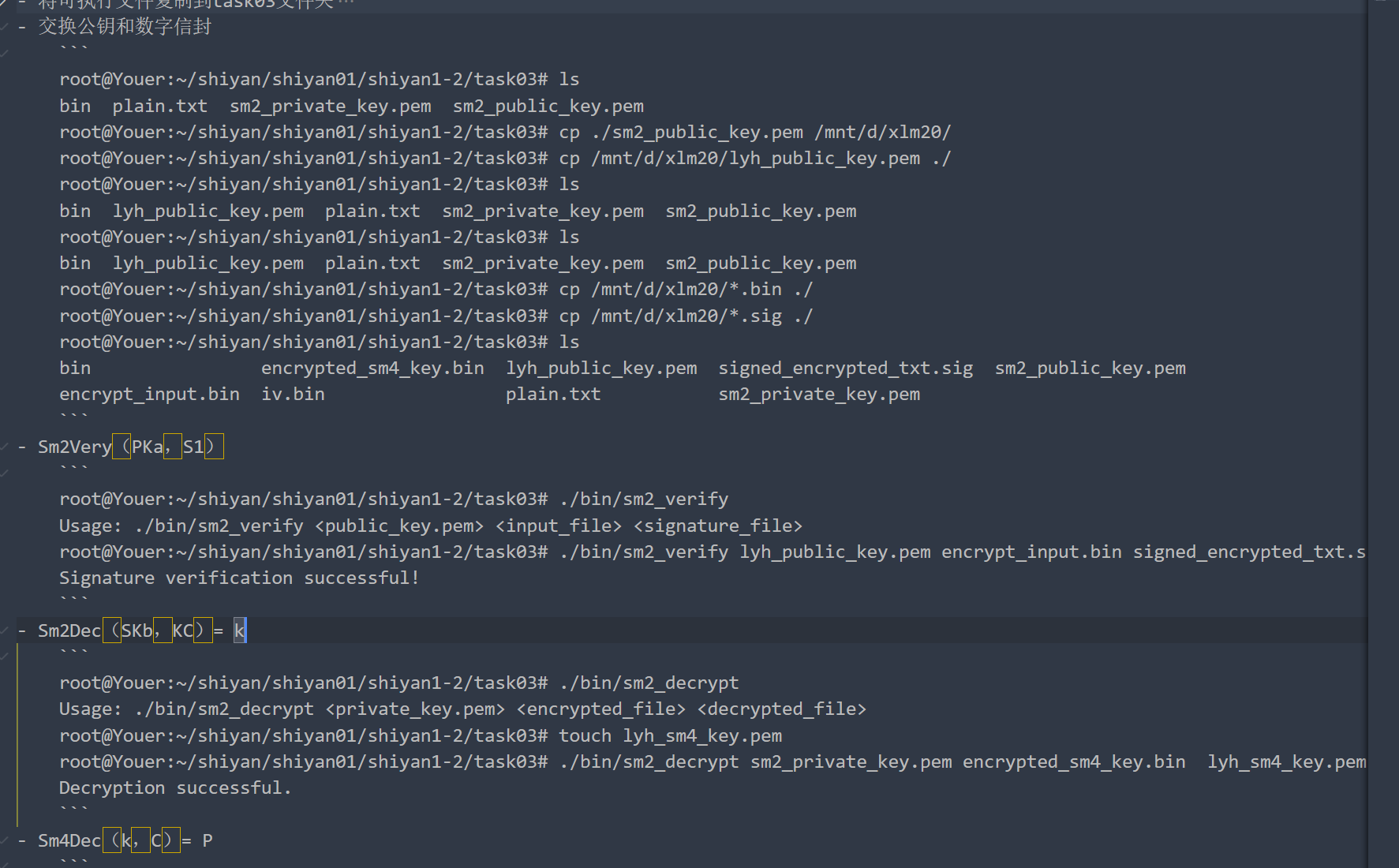
****

图 14 部分过程

**3．实验体会**

**3.1 调试中出现的问题及解决过程**

**3.1.1 OpenSSL中sm2代码解密失败，提示ASN.1编码出错问题**

**报错信息：**

        root@Youer:~/shiyan/shiyan01/shiyan1-2/task01/test\_sm2# ./sm2\_decrypt sm2\_private.pem encrypted\_file.bin decrypted\_file.bin

        OpenSSL error: error:068000A8:asn1 encoding routines::wrong tag

        OpenSSL error: error:0688010A:asn1 encoding routines::nested asn1 error

        OpenSSL error: error:1A800068:SM2 routines::reason(104)

**具体表现：**

* **无法通过解密程序解密，但可以通过openssl命令解密**
* **AI的代码无法解决这一bug，且总是重复一些已经排除的可能**

**解决问题的关键在于明确根本问题，之前尝试解决时猜测了很多解决思路，比如转换密文格式等，但这些只是猜测，最终浪费了自己的时间。这让我想到了**[**“XY问题”**](https://coolshell.cn/articles/10804.html)**：没有去问怎么解决问题X，而是去问解决方案Y应该怎么去实现和操作。**

**同时不要就一个代码问AI太久，AI可能跳不出原有的框架，就会一直给出错误代码。要学会及时开启新对话。**

**3.1.2 GmSSL中SM2私钥格式问题**

**在GmSSL的SM2生成签名时遇到私钥读取失败的问题。在处理过程中发现是私钥格式的问题，但没能很好的总结经验，而是重复询问AI-报错-继续询问的循环。**

**其实在之前就已经实现了SM2加解密的程序，已经有了读取私钥的代码实现。可以直接把相关代码拷贝过来给AI作为参考，这样就可以快速解决这个问题了。**

**3.2 心得体会**

**实验1-1部分还好实验1-2部分即代码编程部分是最难的部分，这一过程简直就是“赛博炼丹”，以我的知识储备来说（特别是刚开始时），很难判断AI的回答正确与否，只能重复“询问-报错-继续询问”的循环，期望某一次能够解决问题。**

**但在这一过程中，我也有一些经验与教训。有些是针对这一次实验的，也有一些是通用的方法。**

**首先是对于调用openssl库编程的部分，一定要知道openssl本身不是直接支持sm2等算法，需要调用EVP接口来实现相关功能。同时其不存在”openssl/sm2.h”等头文件，这些是混用了gmssl的代码。**

**其次是对于调用gmssl库编程的部分，这里与openssl不同，不应该使用evp接口，同时也不存在”gmssl/evp.h”的头文件。同时其错误处理的头文件是”gmssl/error.h”而不是”gmssl/err.h”，不要混用了openssl的代码。**

**总结一下，就是要知道这些库对相关算法的支持形式，与能够分辨容易混淆的代码。**

**同时，也有对AI提问的经验教训的总结。3.1中其实已经提到了一些经验：要明确根本问题，避免“XY问题”；及时开启新对话；利用好已经成功的代码做参考。**

**但还有一些经验需要总结：根据反馈及时调整自己的提问，知道自己在做什么，避免AI重复犯错浪费时间；如果C语言代码不好实现，可以考虑换python实现程序（但要注意实验要求）等。**

**3.3 实验改进建议**

**建议告诉大家调用openssl和gmssl编程时有哪些方法与注意事项，或者早一些给出示例代码。**