实验三 密码模块实现

1-3 学时实践要求（30 分）

1. 阅读学习 《GM/T 0016智能密码钥匙密码应用接口规范》和《GMT 0018 密码设备应用接口规范》
2. 在 Ubuntu或openEuler中（推荐 openEuler）中编译运行附件中《GM/T 0016智能密码钥匙密码应用接口规范》相关代码，并新增完成标准中至少一项功能。使用Markdown记录详细记录实践过程，每完成一项功能或者一个函数git commit 一次。（15分）
3. 在 Ubuntu或openEuler中（推荐 openEuler）中编译运行附件中《GMT 0018 密码设备应用接口规范》相关代码，并新增完成标准中至少一项功能。（15分）
4. 实验记录中提交 gitee 课程项目链接，提交本次实验相关 git log运行结果。
5. 提交要求：

- 提交实践过程Markdown和转化的PDF文件
- 代码，文档托管到gitee或github等
- 记录实验过程中遇到的问题，解决过程，反思等内容，用于后面实验报告
- 实验过程文档GitHub链接

完成过程

裸卡 0018 代码测试

- 查看说明

```
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf# git pull
Already up to date.
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf# tree
.
├── HS_besti_linux_SDK_20200924
│   ├── config
│   │   ├── config.sh
│   │   └── unconfig.sh
│   ├── readMe.txt
│   └── so
│       ├── arm
│       │   ├── libhs_guomi_vpn.so
│       │   └── test
│       ├── example
│       │   ├── Makefile
│       │   ├── sdf.h
│       │   └── test.c
│       ├── include
│       │   └── sdf.h
│       ├── x86
│       │   ├── libhs_guomi_vpn.so
│       │   └── test
│       └── x86_64
│           ├── libhs_guomi_vpn.so
│           └── test
├── config
```

```
│   ├── config.sh
│   └── unconfig.sh
├── example-x86-64.zip
├── examples-arm.zip
├── readMe.txt
└── rochs0018电路图.pdf

9 directories, 19 files
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf# cat readMe.txt
1. 先运行config中的config.sh
cd config
chmod +x *.sh
sudo  ./config.sh

2. 根据自己的平台情况解压exaple

unzip example-x86-64.zip
cd example
make
sudo ./test
```

- 先运行config中的config.sh

```
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf# cd config
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf/config# chmod +x *.sh
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf/config# sudo  ./config.sh
Service udev restarted!
run finished!
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf/config# cd ..
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf# unzip example-x86-64.zip
Archive:  example-x86-64.zip
creating: examples/
inflating: examples/sm4.o
inflating: examples/test.o
inflating: examples/sm4.c
inflating: examples/Makefile
inflating: examples/test.c
inflating: examples/test
inflating: examples/libhsctu_guomi_vpn.so
inflating: examples/sm4.h
inflating: examples/sdf.h
```

- 解压运行出现报错

```
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf#  cd examples
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf/examples# rm test
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf/examples# make
g++ sm4.o test.o  ./libhsctu_guomi_vpn.so -lpthread -o test
root@Youer:~/bestidiocs2024/ch06/rochs0018sdf/examples# sudo ./test
```

```
iterator --------   erorr
open devces fail rv = 0x01000004
```

- 由此需要解决WSL无法连接USB的问题。具体过程参考最后的解决问题板块。
- 最终输出结果

```
open device success!
open session success!
pOutRand:
3128121f e765993e b7b6169b a0447cf6
SDF_GenerateRandom success!
SDF_ImportRootKeyAndDeviceSN success
SN:hs_00000000000014208
CosVer: 4.2.08
ImportKeyPair success
EccBackUpKeyPair success
ExportKeyPair success
SGD_SM3Hash success
SM2EncDec success
SM2SignVer success
SGD_SM1_ECB Encrypt datasize: 4000000 Bytes used time: 1664913 us
SGD_SM1_ECB Encrypt average speed: 19220223 bps
SGD_SM1_ECB Decrypt datasize: 4000000 Bytes used time: 1651084 us
SGD_SM1_ECB Decrypt average speed: 19381206 bps
SM1_ENC_DEC_ECB success.
SGD_SM1_CBC Encrypt datasize: 4000000 Bytes used time: 1711877 us
SGD_SM1_CBC Encrypt average speed: 18692931 bps
SGD_SM1_CBC Decrypt datasize: 4000000 Bytes used time: 1692697 us
SGD_SM1_CBC Decrypt average speed: 18904741 bps
SM1_ENC_DEC_CBC success.
SGD_SM1_OFB Encrypt datasize: 4000000 Bytes used time: 1448136 us
SGD_SM1_OFB Encrypt average speed: 22097372 bps
SGD_SM1_OFB Decrypt datasize: 4000000 Bytes used time: 1446069 us
SGD_SM1_OFB Decrypt average speed: 22128957 bps
SM1_ENC_DEC_OFB success.
SGD_SM4_ECB Encrypt datasize: 4000000 Bytes used time: 1794267 us
SGD_SM4_ECB Encrypt average speed: 17834580 bps
SGD_SM4_ECB Decrypt datasize: 4000000 Bytes used time: 1797626 us
SGD_SM4_ECB Decrypt average speed: 17801255 bps
SM4_ENC_DEC_ECB success.
SGD_SM4_CBC Encrypt datasize: 4000000 Bytes used time: 1872762 us
SGD_SM4_CBC Encrypt average speed: 17087061 bps
SGD_SM4_CBC Decrypt datasize: 4000000 Bytes used time: 1865848 us
SGD_SM4_CBC Decrypt average speed: 17150378 bps
SM4_ENC_DEC_CBC success.
SGD_SM4_OFB Encrypt datasize: 4000000 Bytes used time: 1478838 us
SGD_SM4_OFB Encrypt average speed: 21638610 bps
SGD_SM4_OFB Decrypt datasize: 4000000 Bytes used time: 1465698 us
SGD_SM4_OFB Decrypt average speed: 21832601 bps
SM4_ENC_DEC_OFB success.
SGD_IPSEC_SM1 Encrypt datasize: 4024000 Bytes used time: 2126279 us
```

```
SGD_IPSEC_SM1 Encrypt average speed: 15140063 bps
SGD_IPSEC_SM1 Decrypt datasize: 4024000 Bytes used time: 2128484 us
SGD_IPSEC_SM1 Decrypt average speed: 15124379 bps
SM1_ENC_DEC_IPSEC success.
SGD_IPSEC_SM4 Encrypt datasize: 4024 Bytes used time: 2334 us
SGD_IPSEC_SM4 Encrypt average speed: 13792630 bps
SGD_IPSEC_SM4 Decrypt datasize: 4024 Bytes used time: 2192 us
SGD_IPSEC_SM4 Decrypt average speed: 14686131 bps
SM4_ENC_DEC_IPSEC success.
```

- 借助AI阅读理解代码

龙脉 0016 代码测试

- 前期准备
  - 插入龙脉芯片，参考之前的步骤将USB连接到WSL中
  - 解压老师仓库中的longmaiskf0016-stu.zip压缩包
  - 在samples/skf/linux_mac文件夹中新建lib/linux文件夹，在linux文件夹下把longmaiskf0016-stu\skf\linux\x64文件夹拷贝过来
  - 之后开始make示例
  - 阅读readme.txt文件中的信息，我们可以了解各个示例的作用

```
GM3000 国密应用相关例程

DevAuth     --------------设备认证例程

EncryptData --------------数据加解密例程

RemoteUnblock ------------远程解锁例程

Signature --- 签名验证例程
```

- `encrypt`demo运行(报错且未解决)
  - 将makefile_linux文件重命名为makefile文件
  - 修改makefile文件的LINKFLAGS中的x86改为x64
  - 先执行`make clean`命令，再执行`make`命令

```
root@Youer:~/bestidiocs2024/ch06/longmaiskf0016-
stu/samples/skf/linux_mac/encrypt# make clean
rm -f encryptTest
rm -f *.o encryptTest
root@Youer:~/bestidiocs2024/ch06/longmaiskf0016-
stu/samples/skf/linux_mac/encrypt# make
rm -f encryptTest
g++  -o encryptTest main.o ../lib/linux/x64/libgm3000.1.0.so
```

- 执行`sudo ./encryptTest`命令，发现什么输出都没有
- 阅读`main.cpp`代码，发现如果正常运行应该有运行成功的提示信息
- 修改main.cpp代码，添加更多的提示信息，重新编译运行，出现报错：

```
root@Youer:~/bestidiocs2024/ch06/longmaiskf0016-
stu/samples/skf/linux_mac/encrypt# sudo ./encryptTest
Starting to enumerate devices...
Device enumeration successful.
Trying to connect to the device...
Error: Function call failed at line 47, error code: 167772166
```

  - 结果分析：设备枚举成功，但设备在连接阶段出现通信错误，错误码：167772166
- 询问AI：无合适的解决方法
- 解决问题的尝试：
  - 方向一：尝试去官网找技术文档看看这个错误码是什么原因，但根本没有技术文档或其他帮助文档.....
    - 基本上都需要与工作人员联系。放弃这个方向。
  - 方向二：
    - 考虑到虚拟机可以正常运行，同时其与USB连接方式一般是基于虚拟 USB 控制器的直接连接；而WSL是通过网络(USB/IP协议)进行连接。
    - 但是，WSL可以识别这个USB，`lsusb`命令可以发现这个USB的信息。该USB应该支持USB/IP协议。
    - WSL与USB通信原理
  - 方向三：内核问题：其与`5.15.167.4-microsoft-standard-WSL2`内核不兼容。无法验证思路正确性。
  - 方向四：系统位数问题。尝试修改makefile文件并导入32位的动态库来编译32位程序。但最终结果没有变化。

```
root@Youer:~/bestidiocs2024/ch06/longmaiskf0016-
stu/samples/skf/linux_mac/encrypt# make
g++  -m32 -c -o main.o main.cpp
rm -f encryptTest
g++  -m32 -o encryptTest main.o ../lib/linux/x86/libgm3000.1.0.so
root@Youer:~/bestidiocs2024/ch06/longmaiskf0016-
stu/samples/skf/linux_mac/encrypt# ls
encryptTest  main.cpp  main.o  makefile  makefile_mac
root@Youer:~/bestidiocs2024/ch06/longmaiskf0016-
stu/samples/skf/linux_mac/encrypt# ./encryptTest
Starting to enumerate devices...
Device enumeration successful.
Trying to connect to the device...
Error: Function call failed at line 47, error code: 167772166
```

- 综上，没有解决连接失败的问题

- 同理，其他demo也无法运行

git相关

- 由于本次实验只是编译运行老师发的代码，所以没有单独新建一个仓库，因此没有远程仓库与git记录

问题记录与解决过程

问题一：解决WSL无法连接USB

核心参考文章

- 连接 USB 设备
- WSL2连接USB设备（以USRP B210为例）

具体步骤

- 先决条件
  - 运行 Windows 11（内部版本 22000 或更高版本）
  - 需要具有 x64 处理器的计算机。（x86 和 Arm64 目前不支持 usbipd win）。
  - WSL 已安装并使用最新版本进行设置。
  - Linux 发行版已安装并设置为 WSL 2
- 打开终端管理员，运行下面的命令安装 USBIPD

```
winget install --interactive --exact dorssel.usbipd-win
```

  - 也可以参考连接 USB 设备文章中的其他安装方法
- 附加USB设备
  - 通过以管理员模式打开 PowerShell 并输入以下命令，列出所有连接到 Windows 的 USB 设备。

```
usbipd list
```

```
PS C:\Users\xlm20> usbipd list
Connected:
BUSID  VID:PID    DEVICE                                                        STATE
2-1    096e:0321  USB 大容量存储设备                                            Shared
2-2    10a5:a920  FPC Fingerprint Reader (Disum)                                Not shared
2-7    13d3:5523  2K Camera, IR Camera                                          Not shared
2-10   8087:0026  英特尔(R) 无线 Bluetooth(R)                                   Not shared

Persisted:
GUID                                  DEVICE
beaa4ec8-1cf0-428b-a301-f415c65eba1b  G102 LIGHTSYNC, USB 输入设备, 虚拟 HID 框架(VHF) HID 设备
```

CSDN @Youer0219

  - 在附加 USB 设备之前，必须使用命令 usbipd bind 来共享设备，从而允许它附加到 WSL。 这需要管理员权限。 选择要在 WSL 中使用的设备总线 ID，然后运行以下命令。 运行命令后，请再次使用命令 usbipd list 验证设备是否已共享。

```
usbipd bind --busid 2-1
```

- 这里首先要判断 你要共享的USB是哪个设备总线ID。我这里的是2-1，对应096e:0321 USB大容量存储设备
- 如何判断成功共享：再次运行usbipd list命令。注意最左侧STATE是不是Shared，如果是，说明共享成功

- 继续执行下面的命令usbipd attach --wsl --busid 2-1
  - 这里的2-1要换成自己的设备总线ID
  - 

- 在Ubuntu中运行lsusb命令
  - 没有这个命令就去问AI怎么安装.....
  - 出现下图中间那个说明连接成功

  - 根据你的应用程序，你可能需要配置 udev 规则以允许非根用户访问设备。(我直接是root用户，没有经过这一步骤)
  - 请最好不要关闭WSL命令提示符
  - 同时，请注意，只要 USB 设备连接到 WSL，Windows 将无法使用它。所以如果你在Windows下找不到这个U盘的提示，也可以间接说明你连接成功

- 结果验证
  - 只要出现下面两行说明连接是没有问题的，如果连接失败就是代码的问题

```
open device success!
open session success!
```

    - 代码问题解决方法：检查你的芯片，如果是EU的芯片就需要用老师最新发的那个安装包
  - 成功的结果参考实际过程中的输出

问题二：如何使用老师最新发的代码

- 将老师发的压缩包拷贝到Windows桌面，解压。
- 进入解压后的文件夹，将唯一的文件重命名，加上.zip后缀
- 再次解压，得到最终可用的代码文件夹
- 之后复制到WSL或需要的地方即可

问题三：龙脉设备运行demo时出现设备枚举成功，但在连接阶段出现通信错误的报错（未解决）

- 报错信息：

```
root@Youer:~/bestidiocs2024/ch06/longmaiskf0016-
stu/samples/skf/linux_mac/encrypt# sudo ./encryptTest
Starting to enumerate devices...
Device enumeration successful.
Trying to connect to the device...
Error: Function call failed at line 47, error code: 167772166
```

- 这个问题的解决思路在实践过程中已有充分讲述。目前并未解决该问题。
- 后续如有解决，在此进行论述。