金仕达商密改造指引

商密改造前，客户端接入普通交易网关，通过认证和登录即可接入；而商密改造后，客户端需要接入签名认证网关和 SSL 网关前置，此时要求客户持有对应厂商根证书，通过签名认证网关下载用户证书，然后在持有有效用户证书的情况下，与商密 SSL 网关加密通讯链路，继而再发起登录，成功后方可交易。

◇ 1.头文件与库

本次商密 API 需要引入厂商库，其中 sm*_sdk.dll 提供通讯链路服务，而 sm*_certsdk.dll 则提供证书服务功能，例如证书下载、证书延期等.

新增 ProxyDLL_*.dll 专为商密接入使用（API 接入客户使用）。

在客户第一次登录时，API 会自动下载证书，并且与 SSL 网关建立连接

并且同时提供 sm*_certsdk.dll，sm*_sdk.dll，B2B 客户端也可以自行调用厂商 DLL，与 ssl 网关建立通讯，下载证书管理相关功能等等

商密厂商	头文件	通讯链路服务动态库	证书服务动态库
格尔	SMApi.h	smk_sdk.dll/so	smk_certsdk.dll
信安世纪	SMCertApi.h	smi_sdk.dll/so	smi_certsdk.dll

新增如下 dll:(商密通道接入使用，请随商密厂商的库放在一起使用，API 接入客户使用)

格尔：ProxyDLL_KOAL.(dll/so)

信安世纪：ProxyDLL_INFOSEC.(dll/so)

注意：新增几个动态库需要跟 kstradeapi 放在同一个目录下，如果不需要接入商密，则不需要放到此目录下，发布包中新增 GMSDK 文件，存放商密新增相关文件

◇ 2.商密改造范围

此次商密改造只涉及交易连接部分，行情部分未做改造。行情保持以往使用方式不变。

◇ 3. 术语说明

- **签名认证网关**

签名认证网关为客户端和服务端之间通信提供身份认证功能，保障接入客户端的身份真实性。提供证书生成、协同密钥生成、身份认证功能。

- **SSL 网关**

双向 SSL 通道，业务层数据先到 SSL 网关，再到柜台。

- **根证书**

根证书是一份特殊的证书,它的签发者是它本身,下载根证书就表明您对该根证书以下所签发的证书都表示信任,而技术上则是建立起一个验证证书信息的链条,证书的验证追溯至根证书即为结束。所以说用户在使用自己的数字证书之前必须先下载根证书。

终端厂商打包终端时（由期货公司提供根证书，每家根证书不一样），需要将根证书放入指定路径，一起打包进终端介质，供用户下载。

各厂商根证书路径：

信安：根证书名为 ifsca-cert.cer，“客户端目录\smidata\”

格尔：根证书为 pem 后缀，存放路径：“客户端目录\smkdata\”

- **用户证书**

由 CA 颁发的数字证书，跟设备和用户绑定，用来验证用户身份，从而建立加密通讯通道。同一用户在同一个设备上开多个客户端，也仅需要一张用户证书；反之同一用户在多个设备上，则需要多张证书，每个设备一张。

用户证书的每日下载数量有限制，具体视期货公司而定，开发者应当避免频繁下证。

Windows 用户证书路径示例：

信安：C:\Users\用户\AppData\Roaming\ifs\smidata\

格尔：C:\Users\用户\AppData\Roaming\koal\smkdata\

Linux 用户证书路径示例：

信安：$HOME/ifs/smidata/

格尔：$HOME/koal/smkdata/

- PIN 码

由用户自定义，**长度至少 6 位，与用户证书绑定**，（下载证书时用户自定义输入），用来保护用户证书（证书下载成功后，需要验证证书，此时需要下载时输入的 PIN 码），保存在客户端本地，多次输错会被锁定，锁定时联系管理员处理，管理员删除服务端证书；或者指导用户，删除本地证书后重试登录。或者调用重置 PIN 码方式解除 pin 锁定

◇ 4. 证书处理流程

证书服务动态库，由商密厂商提供，采用统一接口（请注意各自厂商可能数据结构不一样），接口定义详见头文件 SMCertApi.h。

客户第一次登录时，API 会判断证书是否存在，如果不存在则自动下载证书，证书过期，则自动申请延期，如果遇到其他错误，客户端也可以自行下载好证书放在指定位置，再调用 ReqUserLoginSM（新增）接口。

◇ 5. 客户端调用 API 改造

客户端需要改造的地方有两点:

1）RegisterFront 的地址格式，改为商密地址格式，说明如下：

商密地址统一为 gmsm 开头的协议：

地址：gmsm*://SSLHOST:SSLPORT;protocol://CERTHOST:CERTPORT/，示例如下：

信安：示例一：gmsmi://SSLHOST:SSLPORT;protocol://CERTHOST:CERTPORT/

格尔：示例二：gmsmk://SSLHOST:SSLPORT;protocol://CERTHOST:CERTPORT/

其中 CERTHOST 是证书与协同签名地址，即签名认证网关地址；SSLHOST 为通讯链路地址，即 SSL 网关地址；CERTHOST 与 SSLHOST 的值，根据实际方案可以一样也可以不一样。

2）商密通道 客户端登录时，需要调用新的 ReqUserLoginSM 接口， 没有新增 rsp 接口，复用之前登录应答接口 OnRspUserLogin

◇ 6. 其他 API 相关说明

1）商密 API 不需要客户端手工调用客户端认证 ReqAuthenticate 函数，由 API 内部在通讯链路建立成功后，自动调用。

2）客户端自动重连逻辑：证书验证或者协同签名处理失败，返回客户端 RspError，不自动重连；通讯链路连接过程中的错误，以及通讯链路连接上后断线，会自动重连。

3）针对多对多模式的中继系统，采集终端信息时仍需要在调用 ReqUserLoginSM 接口之前调用 RegisterUserSystemInfo 接口。 （客户端认证接口 ReqAuthenticate 已经包在 ReqUserLoginSM 内部）

4) ReqUserLoginSM，授权码为空时，则不走认证流程（ReqAuthenticate），直接走 ReqUserLogin 登录流程

5)如果需要调高联调测试商密日志级别，改动如下：

则需要新增 KSInterM.ini 配置文件，增加如下

[ProxyInfoSec]
debug=20
或者
[ProxyKoal]
debug=20

6）信安环境，ReqUserLoginSM 登录时新增如下字段，请根据期货公司实际环境填写

```
    ServiceID  =  "JSD";
    AppID      =  "APP_SDK_TEST";
    SecretKey  =
```

◇ 7.证书服务 Api 接口示例代码

◇ 7.1.下载证书：

```
void CertEnroll()
{
    /////商密 Api 全局初始化/////
    int rst = SMCertSDK_Init("sdk.log");
    ///全局初始化
    KSI_Encode_Init();
    SMCertSDK_t *m_cert = new SMCertSDK_t;
    SMCertUserConfig_t *m_cfg = new SMCertUserConfig_t;
 strcpy(m_cfg->BrokerID," ");//此值不要填写,如果是信安环境,赋值为 NULL
    strcpy(m_cfg->UserID,"00001");
```

```
    char pwd[256]={0};
    strcpy(m_cfg->Password,"123456");
    KSI_PWD_EnPack(m_cfg->UserID, m_cfg->Password,pwd,sizeof(pwd));
    strcpy(m_cfg->Password, pwd);
    strcpy(m_cfg->Pin,"12qwas");
    m_cfg->CertSocket = -1;
    strcpy(m_cfg->CertHost,"172.1.1.1");
    strcpy(m_cfg->CertPort,"14000");
    m_cfg->TimeoutMs = 5 * 60 * 1000; //5min
    //如果是信安环境，还需要增加对如下几个字段赋值，接入环境时，需要根
据期货公司环境设置
    #ifdef INFOSYS
    m_cfg->ServiceID  =  "JSD";
    m_cfg->AppID      =  "APP_SDK_TEST";
    m_cfg->SecretKey                                                    =
    "Bx3NF8Z9vjadPyqH7Bpz+JINvHUOFGJWMggIXOUJXCVW1geEMAthM/D6EJpcsDY8
    ymXwpDGcAs24PFPc3sD36Q==";
    m_cfg->NoSync     =  0;
    #endif

    /////商密 Api 创建句柄/////
    rst = SMCertSDK_New(m_cfg, m_cert);
    SMCertSDK_CertEnroll(*m_cert);
    ////结束程序运行////
    ////释放句柄/////
    SMCertSDK_Free();
    ////全局初始化清理/////
    SMCertSDK_Clean();
}
```

◇ 7.2.查询证书：

```
void CertQuery()
{
    /////商密 Api 全局初始化/////
    int rst = SMCertSDK_Init("sdk.log");
    ///全局初始化
    KSI_Encode_Init();

    SMCertSDK_t *m_cert = new SMCertSDK_t;
    SMCertUserConfig_t *m_cfg = new SMCertUserConfig_t;
 strcpy(m_cfg->BrokerID,""); //此值不要填写,如果是信安环境,赋值为 NULL
    strcpy(m_cfg->UserID,"00001");
```

```
    strcpy(m_cfg->Password,"123456");
    char pwd[256]={0};
    KSI_PWD_EnPack(m_cfg->UserID, m_cfg->Password,pwd,sizeof(pwd));
    strcpy(m_cfg->Password, pwd);
    strcpy(m_cfg->Pin,"12qwas");
    m_cfg->CertSocket = -1;
    strcpy(m_cfg->CertHost,"172.1.1.1");
    strcpy(m_cfg->CertPort,"14000");
    m_cfg->TimeoutMs = 5 * 60 * 1000; //5min
    //如果是信安环境，还需要增加对如下几个字段赋值，接入环境时，需要根
据期货公司环境设置
    #ifdef INFOSYS
    m_cfg->ServiceID  =  "JSD";
    m_cfg->AppID      =  "APP_SDK_TEST";
    m_cfg->SecretKey                                                    =
    "Bx3NF8Z9vjadPyqH7Bpz+JINvHUOFGJWMggIXOUJXCVW1geEMAthM/D6EJpcsDY8
    ymXwpDGcAs24PFPc3sD36Q==";
    m_cfg->NoSync     =  0;
    #endif

    /////商密 Api 创建句柄/////
    rst = SMCertSDK_New(m_cfg, m_cert);
    int cert_num = 0;
    const SMCert_t *cert_msg = nullptr;
    rst = SMCertSDK_CertQuery(*m_cert, &cert_msg, &cert_num);
    if (rst != SMCERTSDK_ERR_NONE)
    {
        printf("SMCertSDK_CertQuery Error : 0x%X, %s\n", rst,
m_map_errormsg[rst].c_str());
        return;
    }
    else
    {
        printf("SMCertSDK_CertQuery DONE : 0x%X, %s\n", rst,
m_map_errormsg[rst].c_str());
        //////////可能用户名下没有过证书/////////
        if (cert_num > 0)
        {
            for (int i = 0; i < cert_num; i++)
            {
                SMCert_t t = (SMCert_t)cert_msg[i];
                printf("CertID=%s, UserID=%s, DeviceID=%s, CertInfo=%s,
IsCurrent=%d\n",
```

```
                    t.CertID, t.UserID, t.DeviceID, t.CertInfo,
t.IsCurrent);
            }
        }
        else
        {
            printf("User has NO Certificate!!!\n");
        }
    }
        ////结束程序运行////
        ////释放句柄/////
        SMCertSDK_Free();
        ////全局初始化清理/////
        SMCertSDK_Clean();
 }
```

◇ 7.3.重置 PIN：

```
void ResetPin()
{
    /////商密 Api 全局初始化/////
    int rst = SMCertSDK_Init("sdk.log");
    ///全局初始化
    KSI_Encode_Init();

    SMCertSDK_t *m_cert = new SMCertSDK_t;
    SMCertUserConfig_t *m_cfg = new SMCertUserConfig_t;
 strcpy(m_cfg->BrokerID,"");//此值不要填写,如果是信安环境,赋值为 NULL

    strcpy(m_cfg->UserID,"00001");
    strcpy(m_cfg->Password,"123456");
    char pwd[256]={0};
    KSI_PWD_EnPack(m_cfg->UserID, m_cfg->Password,pwd,sizeof(pwd));
    strcpy(m_cfg->Password, pwd);
    strcpy(m_cfg->Pin,"12qwas");
    m_cfg->CertSocket = -1;
    strcpy(m_cfg->CertHost,"172.1.1.1");
    strcpy(m_cfg->CertPort,"14000");
    m_cfg->TimeoutMs = 5 * 60 * 1000; //5min
    //如果是信安环境，还需要增加对如下几个字段赋值，接入环境时，需要根
据期货公司环境设置
    #ifdef INFOSYS
    m_cfg->ServiceID  =  "JSD";
```

```
    m_cfg->AppID      =  "APP_SDK_TEST";
    m_cfg->SecretKey                                                    =
    "Bx3NF8Z9vjadPyqH7Bpz+JINvHUOFGJWMggIXOUJXCVW1geEMAthM/D6EJpcsDY8
    ymXwpDGcAs24PFPc3sD36Q==";
    m_cfg->NoSync     =  0;
    #endif

    /////商密 Api 创建句柄/////
    rst = SMCertSDK_New(m_cfg, m_cert);
    string NewPIN = "1qaz2wsx";
    rst = SMCertSDK_ResetPin(*m_cert, NewPIN.c_str());
    ////结束程序运行////
    ////释放句柄/////
    SMCertSDK_Free();
    ////全局初始化清理/////
    SMCertSDK_Clean();
}
```

◇ 7.4.废弃用户证书：

```
void CertRevoke()
{
    /////商密 Api 全局初始化/////
    int rst = SMCertSDK_Init("sdk.log");
    ///全局初始化
    KSI_Encode_Init();
    SMCertSDK_t *m_cert = new SMCertSDK_t;
    SMCertUserConfig_t *m_cfg = new SMCertUserConfig_t;
  strcpy(m_cfg->BrokerID,"") //此值不要填写,如果是信安环境,赋值为 NULL
    strcpy(m_cfg->UserID,"00001");
    strcpy(m_cfg->Password,"123456");
    char pwd[256]={0};
    KSI_PWD_EnPack(m_cfg->UserID, m_cfg->Password,pwd,sizeof(pwd));
    strcpy(m_cfg->Password, pwd);
    strcpy(m_cfg->Pin,"12qwas");
    m_cfg->CertSocket = -1;
    strcpy(m_cfg->CertHost,"172.1.1.1");
    strcpy(m_cfg->CertPort,"14000");
    m_cfg->TimeoutMs = 5 * 60 * 1000; //5min
    //如果是信安环境，还需要增加对如下几个字段赋值，接入环境时，需要根
据期货公司环境设置
    #ifdef INFOSYS
    m_cfg->ServiceID  =  "JSD";
```

```
    m_cfg->AppID      =  "APP_SDK_TEST";
    m_cfg->SecretKey                                                    =
    "Bx3NF8Z9vjadPyqH7Bpz+JINvHUOFGJWMggIXOUJXCVW1geEMAthM/D6EJpcsDY8
    ymXwpDGcAs24PFPc3sD36Q==";
    m_cfg->NoSync     =  0;
    #endif

    /////商密 Api 创建句柄/////
    rst = SMCertSDK_New(m_cfg, m_cert);
    char *CertID = new char[50];
    cout <<  "请输入证书编号: ";
    cin >> CertID;
    cout << endl;
    rst = SMCertSDK_CertRevoke(*m_cert, CertID);
    ////结束程序运行////
    ////释放句柄/////
    SMCertSDK_Free();
    ////全局初始化清理/////
    SMCertSDK_Clean();
}
```

◇ 7.5.延期本设备用户证书：

```
void CertDelay()
{
    /////商密 Api 全局初始化/////
    int rst = SMCertSDK_Init("sdk.log");
    ///全局初始化
    KSI_Encode_Init();
    SMCertSDK_t *m_cert = new SMCertSDK_t;
    SMCertUserConfig_t *m_cfg = new SMCertUserConfig_t;
  strcpy(m_cfg->BrokerID,"") //此值不要填写,如果是信安环境,赋值为 NULL
    strcpy(m_cfg->UserID,"00001");
    strcpy(m_cfg->Password,"123456");
    char pwd[256]={0};
    KSI_PWD_EnPack(m_cfg->UserID, m_cfg->Password,pwd,sizeof(pwd));
    strcpy(m_cfg->Password, pwd);
    strcpy(m_cfg->Pin,"12qwas");
    m_cfg->CertSocket = -1;
    strcpy(m_cfg->CertHost,"172.1.1.1");
    strcpy(m_cfg->CertPort,"14000");
    m_cfg->TimeoutMs = 5 * 60 * 1000; //5min
```

```
    //如果是信安环境，还需要增加对如下几个字段赋值，接入环境时，需要根
据期货公司环境设置
    #ifdef INFOSYS
    m_cfg->ServiceID  =  "JSD";
    m_cfg->AppID      =  "APP_SDK_TEST";
    m_cfg->SecretKey                                                =
    "Bx3NF8Z9vjadPyqH7Bpz+JINvHUOFGJWMggIXOUJXCVW1geEMAthM/D6EJpcsDY8
    ymXwpDGcAs24PFPc3sD36Q==";
    m_cfg->NoSync     =  0;
    #endif

    /////商密 Api 创建句柄/////
    rst = SMCertSDK_New(m_cfg, m_cert);
    rst = SMCertSDK_CertDelay(*m_cert);
    ////结束程序运行////
    ////释放句柄/////
    SMCertSDK_Free();
    ////全局初始化清理/////
    SMCertSDK_Clean();
}
```

◇ 8.通讯链路服务 Api 接口示例代码

请联系各自商密硬件厂商，提供对应的通讯链路接口实例代码

需要注意的：

建立 SMSDK 时，user_config 中的 Password

```
int createSMSDK()
{
  SMUserConfig_t user_config
  #ifdef INFOSYS
  user_config.BrokerID   =NULL;
  #else
  user_config.BrokerID   ="";
  user_config.UserID    = "00001";
  user_config.Pin       = "12qwas";
  strcpy(user_config.Password,"123456");
  char pwd[256]={0};
  KSI_PWD_EnPack(m_cfg->UserID, m_cfg->Password,pwd,sizeof(pwd));
  strcpy(user_config.Password, pwd);
  /* config mauth info for download cert */
  user_config.CertSocket =  -1;
  user_config.CertHost   = "172.1.1.1";
  user_config.CertPort   = 14000;
```

```
    user_config.TimeoutMs  = 5 * 60 * 1000; //5min
    #ifdef INFOSYS
    user_config.ServiceID  =  "JSD";
    user_config.AppID      =  "APP_SDK_TEST";
    user_config.SecretKey                                                    =
"Bx3NF8Z9vjadPyqH7Bpz+JINvHUOFGJWMggIXOUJXCVW1geEMAthM/D6EJpcsDY8ym
XwpDGcAs24PFPc3sD36Q==";
    user_config.NoSync     =  0;
    #endif
    nRet = SMSDK_New(&user_config, &sdk);
    return nRet;
}
```

◇ 9. B2B 通讯协议厂商接入注意点：

1. 如果通过商密方式接入下证书以及创建 SSL sdk（SMSDK_New）时，密码需要按照金仕达加密后传入（非商密的接入方式不需要加密密码），登录功能还用原始密码，加密代码如下：

```
static char MMEnum[64] =
    {'!','$','0','1','2','3','4','5','6','7','8','9','A','B','C','D',
    'E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T',
    'U','V','W','X','Y','Z','a','b','c','d','e','f','g','h','i','j',
    'k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z'};
#define MAX_PWD_COLUMN_LEN  64
static char MMReve[128];
static bool g_MMInited=false;
void MMInit()
{
    if (g_MMInited == false) {
        srand((unsigned)time(NULL));
        memset(MMReve,0,sizeof(MMReve));
        for (int i=0;i<64;i++)
        {
            MMReve[MMEnum[i]] = i;
        }
        g_MMInited = true;
    }
}
```

```
//一个进程全局初始化一次
void KSI_Encode_Init()
{
	MMInit();
}
static int MMRandom(int r)
{
	int ret = 0;
	if (r != 0)
		ret = rand() % r;
	return ret;
}

static void One2(char *cc,char c)
{
	*cc = MMEnum[((MMRandom(4)<<4) | ((c>>4) & 0x0f)) & 0xff];
	cc++;
	*cc = MMEnum[((MMRandom(4)<<4) | (c & 0x0f)) & 0xff];
}
char *MMEncodeStr(char *dest,const char *src)
{
	int len = strlen(src);
	if (len > 255)
		len = 255;
	One2(dest,len);
	dest += 2;
	int i;
	for (i=0;i<len;i++,dest+=2)
	{
		One2(dest,src[i]);
	}
	for (int j=(i+1)*2;j<30;j++,dest++)
	{
		*dest = MMEnum[MMRandom(64)];
	}
	*dest = 0;
	return dest;
}
```

//调用这个函数对密码加密：

//入参 khh： 客户号

//入参 pwd1： 原始密码

//入参 pack： 加密成功后，加密后的值

//入参 nSize： 传入的加密后数值的长度

//返回值为 0，则加密成功，否则加密失败

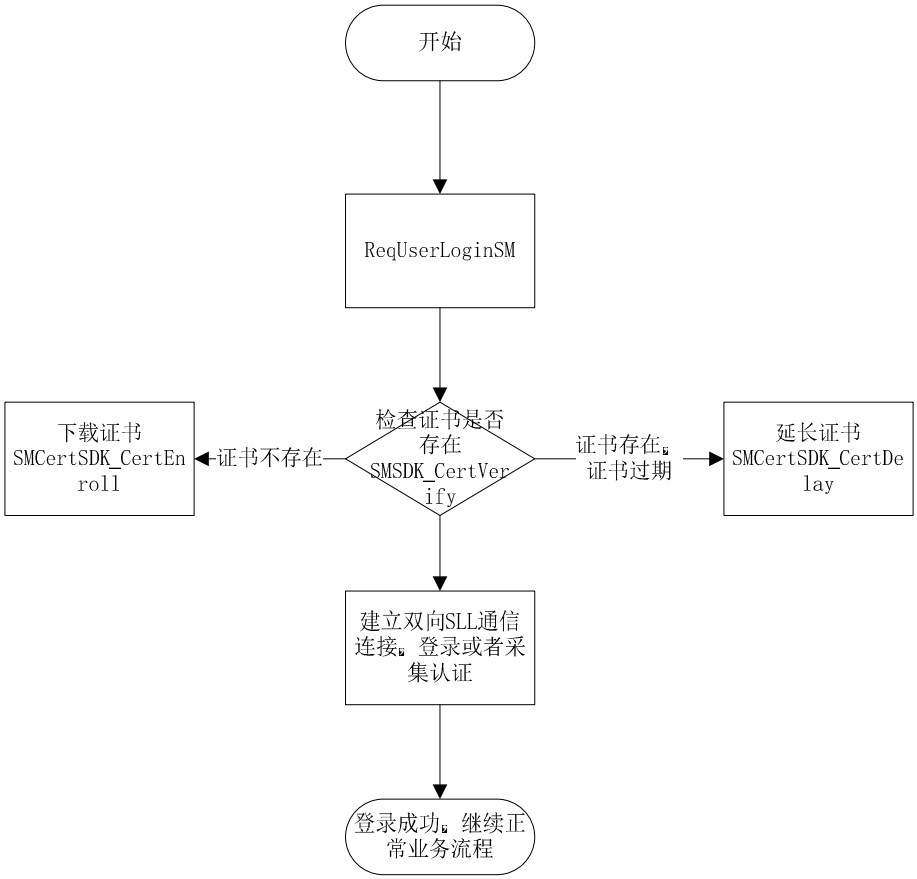
```
int KSI_PWD_EnPack(const char *khh, const char *pwd1, char *pack, int nSize)
{
#define CHECK_STR(str, len, maxlen) { \
                len = 0; \
                if (str) { \
                      len = strlen(str); \
                      if (len > maxlen) return -2; \
                      if (strchr(str, '|')) return -3; \
                } \
           }
      int pwd1_len;
      int khh_len;
      CHECK_STR(pwd1, pwd1_len, MAX_PWD_COLUMN_LEN);
      CHECK_STR(khh, khh_len, MAX_PWD_COLUMN_LEN);
#undef CHECK_STR
      char buf[300];
      char buffer[1024];
      const char *pResult;
      int ResultLen;
      sprintf(buf, "%s|%s|%s|%s",
                  "",
                  khh?khh:"",
                  pwd1?pwd1:"",
                  "" );
      buffer[0] = 'K';
      buffer[1] = 'S';
      buffer[2] = '1';
      buffer[3] = '|';
      MMEncodeStr(buffer + 4, buf);
      pResult = buffer;
      ResultLen = strlen(buffer);
      if (ResultLen >= nSize)
           return ResultLen+1;
      strcpy(pack, pResult);
      return 0;
}
```

2. 中继模式（API）接入时，用户证书，根证书需要放在中继网关所在机器上。

3. 中继模式（B2B 通讯协议）接入时，商密通讯层以及下载证书环节，需要各自终端厂商自行开发，请注

意密码需要加密传入，业务处理逻辑不受影响。

◇ 10.API 商密接入流程图：



◇ 11.ReqUserLoginSM 接口说明：

1. 参数说明

字段类型	字段名称	含义	是否可作为过滤条件
TThostFtdcDateType	TradingDay	交易日	无
TThostFtdcBrokerIDType	BrokerID	经纪公司代码	无
TThostFtdcUserIDType	UserID	用户代码	必填
TThostFtdcPasswordType	Password	密码	必填
TThostFtdcProductInfoType	UserProductInfo	用户端产品信息	无

TThostFtdcProductInfoType	InterfaceProductInfo	接口端产品信息	无
TThostFtdcProtocolInfoType	ProtocolInfo	协议信息	无
TThostFtdcMacAddressType	MacAddress	Mac 地址	必填
TThostFtdcPasswordType	OneTimePassword	动态密码	无
TThostFtdcIPAddressType	ClientIPAddress	终端 IP 地址	必填
TThostFtdcLoginRemarkType	LoginRemark	登录备注	无
TThostFtdcIPPortType	ClientIPPort	终端 IP 端口	无
TThostFtdcAuthCodeType	AuthCode	认证码	必填
TThostFtdcAppIDType	AppID	App 代码	必填
TThostFtdcPasswordType	PIN	PIN 码	必填

信安新增如下字段，请根据期货公司实际情况赋值

TThostFtdcAppIDType	MAuthServiceID;	信安必填，格尔不填
TThostFtdcAppIDType	MAuthAppID;	信安必填，格尔不填
TThostFtdcSecretKeyType	MAuthSecretKey;	信安必填，格尔不填

2. 使用说明：

如调用 ReqUserLoginSM 返回为-5，表示当前连接上的是非商密前置，请继续原非商密代码流程，请一定要根据返回值做必要的处理。

3. 错误码有关证书 SSL 连接部分说明：

错误代码	错误含义
4100	SSL 连接过程中的错误
4101	下证过程中，校验用户名密码失败
4102	下证过程中，其他错误
4103	下载证书超时
4104	登录错误
4105	SSl 连接超时
4106	证书连接错误
4107	登录过程中，证书不存在
4108	登录过程中，证书过期
4109	PIN 码错误
4110	PIN 码被锁
4111	. 商密 API 加载失败

错误代码说明：

errorid=4110:pin 码如果连续输入错误多次会被锁定，此时需要重置 pin 码，错误阈值由期货公司设置

errorid=4111:程序目录下没有对应的 sm*_sdk.dll 或者 ProxyDLL_*.dll