

莱德物联网接口定义---2020.07.24

更新说明:

2020-08-11:

1. writeSetChargeAPDU 添加充值次数

SEUID#SESM2Key#SEtempSm2Key#RandStr#chargeMoney#chargetimes

2. 修正上报剩余金额为负数的情况

3. 增加更新SE种卡密钥接口

更新卡密钥由表端主动给发起请求, 经过外部认证后, 下发更新指令。解析数据 若类型为 MeterRequestExAuth 则根据上报内容调用writeSetEXAuthAPDU生成外部认证的指令并下发, 若解析数据为MeterRequestUpdateCardKey则调用writeGetUpdateCardKeyADPU生成修改卡密钥的指令并下发, 表成功后会正常上报, 无对应回应。使用py脚本为: exauth,

更新为正式密钥调用updateKey2704,updateKey2705,updateKey2706,

更新为测试密钥调用updateKey2704_1,updateKey2705_1,updateKey2706_1,

2020-08-14:

1. 增加 writeGetMeterPulseInfo 读取脉冲记录, 及对应解析。

2. 移除外部认证指令 writeSetEXAuthAPDU, 更新SE中卡密钥无需外部认证, 直接对39 writeGetUpdateCardKeyADPU所属上报回应即可

1.ParseData(解析上报数据)

a.表上报数据 (R)

```
/**
 * 解析上报数据
 *
 * @param Encrytype 是否AES加密: 0为不加密 1 为加密
 * @param Hexstring 表上报数据
```

```

        * @return
        {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
        {"reportDataVersionCode":"01","randStr":"0932AD470689B7B7","mcuVersionCodeString":
        "20190821","cardbuytimes":0,"Result":"SUCCESS","Type":"MeterReport","calcuGas"
        :0,"remainMoney":20,"price":1.98,"buytimes":0,"meterState":
        {"DethMeterOneFlag":"0","DethMeterTwoFlag":"0","MagneticFlag":"0","ForceSafeCheckFlag":"0","ForceScrapFlag":"0","ValveState":"0","OverflowFlag":"0","OutBatteryLowPowerFlag":"0","OutBatteryLosePowerFlag":"0","InnerBatteryLowPowerFlag":"0","TimeErrFlag":"0","LackOfGasFlag":"0","AlarmAndCloseValve":"0","ForceSafeCheckAdviseFlag":"0","ForceScrapAdviseFlag":"0","WireCloseValveState":"0"},"batteryState":"5.0","meterTemperature":23.63,"CSQ":17,"signalPower":-853,"SNR":123,"ECL":0,"CellID":151321118,"PCI":12,"EARFCN":3738,"repFlag":
        {"reportByDeathMeterTwo":"0","reportByDeathMeterOne":"0","reportByMagneticPro":"0","reportByReset":"0","reportByLowPower":"0","reportByLostPower":"0","reportByKey2":"0","reportByKey1":"1","reportByTime":"0"},"cmcState":"0000","meterTime":"20190826133059"}}
        */

```

public static String ParseData(int Encrytype, String Hexstring) .

参数说明:

mcuVersionCodeString: MCU版本号

randStr: SE上报随机数 (上报数据版本为01时)

reportDataVersionCode: 上报数据版本

cardbuytimes: 卡购气次数

calcuGas: 累计用气量

remainMoney: 剩余金额字节

price: 当前单价

buytimes: 无线购气次数

meterState: 仪表状态

DethMeterOneFlag: 死表状态 1 标志

DethMeterTwoFlag: 死表状态 2 标志

MagneticFlag: 磁干扰标志

ForceSafeCheckFlag: 强制安检标志

ForceScrapFlag: 强制报废标志

ValveState: 阀门开关标志, 1-阀门关闭 0-阀门打开

OverflowFlag: 过流标志

OutBatteryLowPowerFlag: 外部电池低电压标志

OutBatteryLosePowerFlag: 外电掉电标志

InnerBatteryLowPowerFlag: 内部电池低电压标志

TimeErrFlag: 时钟错误标志

LackOfGasFlag: 仪表欠量标志

AlarmAndCloseValve: 报警关阀标志

ForceSafeCheckAdviseFlag: 强制安检预警标志

ForceScrapAdviseFlag: 强制报废预警标志

WireCloseValveState: 无线关阀标志

batteryState: 电池状态

meterTemperature: 表内温度

CSQ: 信号强度

signalPower: Signal Power

SNR: SNR

ECL: ECL

CellID: Cell ID

PCI: PCI

EARFCN: EARFCN

repFlag: 上报标志

reportByTime: 定时上报

reportByKey1: 按键1上报

reportByKey2: 按键2上报

reportByLostPower: 掉电上报
reportByLowPower: 低电上报
reportByReset: 复位上报
reportByMagneticPro: 磁保护上报
reportByDeathMeterOne: 死表1上报
reportByDeathMeterTwo: 死表2上报
cmcState: 通讯标志
meterTime: 表内时钟

```
/**
 * 卡表绑定上报
 * @param
 * @return
 */
{"Result": "SUCCESS", "CodeNumber": "20190008880007", "SerialNo": 1, "Data":
{"Type": "MeterBindCard", "Result": "SUCCESS", "MeterOpenAccountState": "00", "userCardNoString": "0000000000", "moduleType": "BC28JA-02-STD", "moduleVersionType": "BC28JAR01A01_ONT", "moduleIMEI": "867726031007158", "cardIMSI": "460040944907093", "cardICCID": "898602b9261771857093", "cardbuytimes": 0, "reportDataVersionCode": "01", "randStr": "0932AD470689B7B7", "mcuVersionCodeString": "20190821", "calcuGas": 0, "remainMoney": 0, "price": 2, "buytimes": 0, "meterState": {"DethMeterOneFlag": "0", "DethMeterTwoFlag": "0", "MagneticFlag": "0", "ForceSafeCheckFlag": "0", "ForceSraptFlag": "0", "ValveState": "1", "OverFlowFlag": "0", "OutBatteryLowPowerFlag": "0", "OutBatteryLosePowerFlag": "0", "InnerBatteryLowPowerFlag": "0", "TimeErrFlag": "0", "LackOfGasFlag": "1", "AlarmAndCloseValve": "0", "ForcesafeCheckAdviseFlag": "0", "ForceScraptAdviseFlag": "0", "WireCloseValveState": "0"}, "batteryState": "5.0", "meterTemperature": 24.22, "CSQ": 20, "signalPower": -807, "SNR": 116, "ECL": 0, "CellID": 151321118, "PCI": 12, "EARFCN": 3738, "repFlag": "0000000000001000", "cmcState": "0000", "meterTime": "20190822173150"}}
***/
```

public static String ParseData(int Encrytype, String Hexstring)

参数说明:

MeterOpenAccountState: 0 未开户, 1 已开户。

userCardNoString: 卡号

moduleType: 模组型号

moduleVersionType: 模组版本号

moduleIMEI: 模组 IMEI

cardIMSI: 卡 IMSI

cardICCID: 卡 ICCID

cardbuytimes: 卡购气次数

mcuVersionCodeString: MCU版本号

randStr: SE上报随机数 (上报数据版本为01时)

reportDataVersionCode: 上报数据版本

cardbuytimes: 卡购气次数

calcuGas: 累计用气量

remainMoney: 剩余金额字节

price: 当前单价

buytimes: 无线购气次数

meterState: 仪表状态 (同上報)

batteryState: 电池状态

meterTemperature: 表内温度

CSQ: 信号强度

signalPower: Signal Power

SNR: SNR

ECL: ECL

CellID: Cell ID

PCI: PCI

EARFCN: EARFCN

repFlag: 上报标志 (同上報)

cmcState:通讯标志
meterTime:表内时钟

b.根据系统内容对表进行回复

- 1.正常/异常上报 根据是否有待下发指令回应 无则回应 断开
- 2.绑定上报 根据系统中卡号对应 确认正确后下发 确认绑定指令

说明：表开户绑定成功前上报此数据，若绑定成功则回复此命令以确认绑定成功；若绑定失败则回复失败或直接发送其它命令。如表端存在SE加密芯片，则需要根据业务系统数据库判断该表是否已协商密钥，未协商则需要下发 密钥协商请求指令 开始密钥协商流程。如表端存在SE加密芯片，则需要根据业务系统数据库判断该表是否已协商过密钥，来判断该条上报指令的加密类型，是否需要经过加密机KMS解密。

```
/*  
 *  
 *  
 * @param Encrytype  
 * @param CodeNumber  
 * @param SerialNo  
 * @param JSON    '{"bindResult':0}";    0成功 1失败  
 * @return  
 */  
public static String writeConfirBindResult(int Encrytype,String CodeNumber, int  
SerialNo, String JSON)
```

2.WriteGetSEKey(下发密钥协商请求)(数据域明文)

a. 下发 (S)

```
/*  
 * 下发密钥协商请求  
 * @param CodeNumber 条码号 14位  
 * @param SerialNo 指令序号  
 * @param jSON "" 空  
 * @return  
 {"Result":"SUCCESS","CodeNumber":"20190008880003","SerialNo":1,"CMDHEXString":"6  
83020190008880003090003D000014116"}  
 */  
public static String writeGetSEKey(String CodeNumber, int SerialNo, String  
JSON)
```

b. 回应(R) (数据域明文)

```

    /**
     * 解析
     * @param isEncry 是否AES加密 0
     * @param Hexstring 表回应数据
     * @return
     {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
     {"Type":"MeterReplyRequestSessionKey","Result":"SUCCESS","SEUID":"196E4204000000",
     "SESM2Key":"E23392C0BC4073194A05388B43701C538652FF396FD42E681037D83134751463
     DC64BF2F968FF23371DB6A9D05FBA2882149D33DA6A668B7271F939D5421DE5A","SEtempSm2Key":
     "C758D7B1AD9BC9F22930D31CCF636CB520ADDC37A34E44B070F0001FC4B4110C
     C5C8E749645E36F9415CD502BEBE184A607C311B17E5506C2D1D6524B446E755","RandStr":"093
     2AD470689B7B7"}}
     */
    public static String ParseData(int isEncry,String Hexstring)

```

参数说明:

SEUID: SE ID 作为SE标识

SESM2Key:SE 固定SM2 公钥

SEtempSm2Key: SE 临时Sm2 公钥

RandStr :随机数

(前三个参数需数据库存储,以备后需加解密需要,随机数具有 时效性,可不存储)

3.WriteSetSessionKey(下发写入协商密钥,传入参数, 内部调用KMS接口)

a. 下发(S)

```

    /**
     * 写入协商密钥
     * @param CodeNumber 条码号
     * @param SerialNo 指令序号
     * @param cmdParams SE相关参数 SEUID#SESM2Key#SEtempSm2Key#RandStr
     * @return
     {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"CMDHEXString":"6
     83020190008880007040020D0010184D4000024c62c04fd781986ab22173e1b37697a8af16e3e8a6
     d83f00310a6a1b91230326841bb7b5084D4000024ebb1c58ac3d1ef42fbf9270eff404567f16e3e8
     a6d83f00310a6a1b912303268114468b89F16"}
     */
    public static String WriteSetSessionKey(String CodeNumber, int SerialNo,
    String cmdParams)

```

b. 回应(R) (数据域明文)

```

    /**
     * 解析
     * @param isEncry 是否AES加密 0
     * @param Hexstring 表回应数据
     * @return
     {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
     {"Type":"MeterReplySetSessionKey","Result":"SUCCESS"}}
     */
    public static String ParseData(int isEncry,String Hexstring)

```

参数说明:

4. WriteSet_SessionKey(下发写入协商密钥, 传入两条APDU指令)

a. 下发 (S)

```
/**
 * 写入协商密钥
 * @param CodeNumber 条码号
 * @param SerialNo 指令序号
 * @param cmdParams
 {"EN_APDU":"84D4000024c62c04fd781986ab22173e1b37697a8af16e3e8a6d83f00310a6a1b912
 30326841bb7b50","DE_APDU":"84D4000024ebb1c58ac3d1ef42fbf9270eff404567f16e3e8a6d8
 3f00310a6a1b912303268114468b89"}
 * @return
 {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"CMDHEXString":"6
 83020190008880007040020D0010184D4000024c62c04fd781986ab22173e1b37697a8af16e3e8a6
 d83f00310a6a1b91230326841bb7b5084D4000024ebb1c58ac3d1ef42fbf9270eff404567f16e3e8
 a6d83f00310a6a1b912303268114468b89F16"}
 */
public static String writeSetSessionKey(String CodeNumber, int SerialNo,
String cmdParams)
```

b. 回应(R) (数据域明文)

```
/**
 * 解析
 * @param isEncry 是否AES加密 0
 * @param Hexstring 表回应数据
 * @return
 {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
 {"Type":"MeterReplySetSessionKey","Result":"SUCCESS"}}
 */
public static String ParseData(int isEncry,String Hexstring)
```

参数说明:

5. WriteSetChargeAPDU(下发充值)(传入参数, 充值金额, 内部调用KMS接口)

a. 下发 (S)

```

    /**
     * 写购入金额(充值)
     * @param CodeNumber 条码号
     * @param SerialNo 指令序号
     * @param cmdParams SE相关参数
SEUID#SESM2Key#SEtempSm2Key#RandStr#chargeMoney#chargetimes
     * @return
{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"CMDHEXString":"6
83020190008880007040020D0100184320210080000300C40D5038E80000000000000000000000
00000009F16"}
     */
    public static String (String CodeNumber, int SerialNo, String cmdParams)

```

b. 回应(R) (充值结果以表端回应数据为准)

```

    /**
     * 解析
     * @param Hexstring
683020190008880007840010d64d688945c277d47d988cc9124930d52c16
     * @return
{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyCharge","Result":"SUCCESS","RemainMoney":2560,"ChargeResult
":0}}
     */
    public static String ParseData(int isEncry,String Hexstring)

```

参数说明:

ChargeResult:0: 充值成功, 其他值: 充值失败。

RemainMoney: 剩余金额, 以此作为充值依据

6. WriteSetCharge_APDU(下发充值)(外部调用KMS接口, 直接传入充值指令组帧)

a. 下发 (S)

```

    /**
     * 写购入金额(充值)
     * @param CodeNumber 条码号
     * @param SerialNo 指令序号
     * @param JsonParams {"ChargeAPDU":"84320210080000300C40D5038E"} 外部调用
接口是需注意 金额* 10000: Integer.toHexString((int) (Money * 10000))
     * @return
{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"CMDHEXString":"6
83020190008880007040020D0100184320210080000300C40D5038E80000000000000000000000
00000009F16"}
     */
    public static String (String CodeNumber, int SerialNo, String JsonParams)

```

b. 回应(R) (充值结果以表端回应数据为准)

```

    /**
     * 解析
     * @param isEncry 0
     * @param Hexstring
    683020190008880007840010d64d688945c277d47d988cc9124930d52c16
     * @return
    {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
    {"Type":"MeterReplyCharge","Result":"SUCCESS","RemainMoney":2560,"ChargeResult":
    0}}
    */
    public static String ParseData(int isEncry,String Hexstring)

```

参数说明:

ChargeResult:0: 充值成功, 其他值: 充值失败。

RemainMoney:剩余金额, 以此作为充值依据

7. WriteValveControl(阀门控制)

a. 下发(S)

```

    /**
     * 阀门控制
     * @param Encrytype
     * @param CodeNumber
     * @param SerialNo
     * @param JSON      '{"ControlType':0}" ControlType 0--开阀 1--关阀 2--取消阀
    门控制
     * @return
    {"Result":"SUCCESS","CodeNumber":"20190008880003","SerialNo":1,"CMDHEXString":"6
    8302019000888000304001048d3416444F1c12315691F2E5286FA9B8916"}
     */
    public static String writeValveControl(int isEncry,String CodeNumber, int
    SerialNo, String jSON)

```

b. 回应(R)

```

    /**
     * 解析
     * @param isEncry
     * @param Hexstring
    683020190008880007840010daa515c84f0ec89921d5a296058712341616
     * @return  {"CodeNumber":"20190008880007","SerialNo":1,"Data":
    {"Type":"MeterReplywriteValveControl","Result":"SUCCESS","MeterState":"0000"}}
    MeterState
     */
    public static String ParseData(int isEncry,String Hexstring)

```


参数说明:

MeterState:表状态

8.WriteTime(校时)

a. 下发(S)

```
/**
 * 校时
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON      '{"CheckTime':'20190711122222'}' //yyyyMMDDHHmmSS
 * @return
 {"Result":"SUCCESS","CodeNumber":"20190008880003","SerialNo":1,"CMDHEXString":"6
 8302019000888000304001048D3416444F1C12315691F2E5286FA9B8916"}
 */
public static String WriteTime(int Encrytype,String CodeNumber, int
SerialNo, String jSON)
```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring
 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"CodeNumber":"20190008880007","SerialNo":1,"Data":
 {"Type":"MeterReplyWriteTime","Result":"SUCCESS"}}
 */
public static String ParseData(int Encrytype,String Hexstring)
```

9.WriteSetPrice(设置价格)

a. 下发(S)

```
/**
 * 设置价格
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
```

```

    * @param JSON
    {'priceType':1,'isChangePrice':1,'priceEffectDate':'19082201','priceDateCount':4
    , 'priceDetailCount':2, 'priceDate':[{ 'Month':'1', 'Day':2, 'PriceType':'0'},
    { 'Month':'6', 'Day':15, 'PriceType':'1'}, { 'Month':'7', 'Day':26, 'PriceType':'0'},
    { 'Month':'9', 'Day':19, 'PriceType':'1'}], 'PriceDetail':
    [{ 'PriceOne':1.23, 'GasOne':11, 'PriceTwo':1.34, 'GasTwo':12, 'PriceThree':1.45, 'NewP
    riceOne':1.56, 'NewGasOne':101, 'NewPriceTwo':1.67, 'NewGasTwo':102, 'NewPriceThree
    ':1.78},
    { 'PriceOne':2.21, 'GasOne':21, 'PriceTwo':2.32, 'GasTwo':22, 'PriceThree':2.43, 'NewP
    riceOne':2.54, 'NewGasOne':201, 'NewPriceTwo':2.65, 'NewGasTwo':202, 'NewPriceThree
    ':2.76}]]}"
    * @return
    {"Result":"SUCCESS","CodeNumber":"20190008880003","SerialNo":1,"CMDHEXString":"6
    8302019000888000304001048D3416444F1C12315691F2E5286FA9B8916"}
    */
    public static String WriteSetPrice(int Encrytype,String CodeNumber, int
    SerialNo, String jSON)
    参数说明:
    * priceType: 单价类型 0 普通单价, 1 阶梯单价
    * isChangePrice: 新单价修改标志 0 不启用, 1 启用
    * priceEffectDate: 单价生效日期
    * priceDateCount: 阶梯日期个数 (最多14个)
    * priceDetailCount: 阶梯价格套数 (最多4个)
    * priceDate: { 'Month':'1', 'Day':2, 'PriceType':'0'} 表示 1月2号启用索引为0的阶梯
    * PriceDetail:
    { 'PriceOne':1.23, 'GasOne':11, 'PriceTwo':1.34, 'GasTwo':12, 'PriceThree':1.45, 'NewP
    riceOne':1.56, 'NewGasOne':101, 'NewPriceTwo':1.67, 'NewGasTwo':102, 'NewPriceThree
    ':1.78} 单价1 用量1 单价2 用量2 单价3 新单价1 新气量1 新单价2 新气量2 新单价三

```

b. 回应(R)

```

    /**
    * 解析
    * @param Hexstring
    6830201900088800078400109424a0703ef2d4054993ffe89e9d608eb916
    * @return {"CodeNumber":"20190008880007","SerialNo":1,"Data":
    {"Type":"MeterReplyWriteSetPrice","Result":"SUCCESS"}}
    */
    public static String ParseData(int Encrytype,String Hexstring)

```

10.WriteGetPrice(读价格)

a. 下发(S)

```

/**
 * 读价格
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON      ""
 * @return
 {"Result":"SUCCESS","CodeNumber":"20190008880003","SerialNo":1,"CMDHEXString":"6
 83020190008880007010010c8d669033fb478c007778dadf58f7f51ba16"}
 */
public static String WriteGetPrice(int Encrytype,String CodeNumber, int
SerialNo, String jSON)

```

b. 回应(R)

```

/**
 * 解析
 * @param Hexstring
 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"CodeNumber":"20190008880007","SerialNo":1,"Data":
 {"Type":"MeterReplyPriceDetail","Result":"SUCCESS","priceType":0,"isChangePrice"
 :0,"priceEffectDate":"19082110","priceDateCount":0,"priceDate":
 [],"priceDetailCount":1,"PriceDetail":
 [{"PriceOne":1.98,"GasOne":20,"PriceTwo":2.38,"GasTwo":40,"PriceThree":2.97,"New
 PriceOne":2.1,"NewGasOne":0,"NewPriceTwo":0,"NewGasTwo":0,"NewPriceThree":0}]}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明：(同设置价格参数)

```

11.WriteGetParam(读参数)

a. 下发(S)

```

/**
 * 读参数
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON      ""
 * @return
 {"Result":"SUCCESS","CodeNumber":"20190008880003","SerialNo":1,"CMDHEXString":"6
 8302019000888000304001048d3416444F1C12315691F2E5286FA9B8916"}
 */
public static String writeGetParam(int Encrytype,String CodeNumber, int
SerialNo, String jSON)

```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring
683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyGetParam","Result":"SUCCESS","isOverflow":0,"overflowParam":0
,"ControlVersionStr":"A5","CodeVersionStr":"V1.9","remainMoney":60,"calcuGas":0,
"calcuMoney":60,"buytimes":0,"price":1.56,"meterTimeString":"190822105516","unUs
eGasDay":0,"unUserGasSec":0,"deathMeterFlag":0,"deathMeterOne":10,"deathMeterTwo
":3,"limitBuyFlag":0,"limitBuyMoney":500,"UserCardNo":"1111111111","isOpenAccoun
t":1,"FacortyCode":34,"calcuStairGas":0,"forceSafeCheckDays":1,"forceScrapDays":
1,"setForceSafeCheckYear":3,"setForceScrapYear":12,"setAlarmValueOne":5,"setAlar
mValueTwo":0,"totalOverUse":0,"curroverUse":0,"handleOverUseDay":"000000"}}
 */
```

```
public static String ParseData(int Encrytype,String Hexstring)
```

参数说明:

isOverflow: 过流标志 0 未过流, 1 已过流

overflowParam: 过流参数

ControlVersionStr+ CodeVersionStr: 控制器版本号

remainMoney: 剩余金额

calcuGas: 累计用气量

calcuMoney: 累计购气金额

buytimes: 购气次数

price: 当前单价

meterTimeString: 表时间

unUseGasDay: 无用气天数

unUserGasSec: 无用气秒数

deathMeterFlag: 死表使能

deathMeterOne: 死表 1 判定天数

deathMeterTwo: 死表 2 判定天数

limitBuyFlag: 限购使能

limitBuyMoney: 限购金额

UserCardNo: 用户卡号

isOpenAccount: 开户标志

FacortyCode: 厂家代码

calcuStairGas: 当前阶梯用量

forceSafeCheckDays: 强制安检计数 (单位: 天)

forceScrapDays: 强制报废计数 (单位: 天)

setForceSafeCheckYear: 设定安检时间 (单位: 年)

setForceScrapYear: 设定报废时间 (单位: 年)

setAlarmValueOne: 设定报警金额 1 (单位: 方)

setAlarmValueTwo: 设定报警金额 2 (单位: 方)

totalOverUse: 总透支金额

curroverUse: 当前透支金额

handleoverUseDay: 处理透支时间

12.WriteSetParam(设置参数)

a. 下发(S)

```

    /**
     * 设置参数
     *
     * @param Encrytype
     * @param CodeNumber
     * @param SerialNo
     * @param JSON
     */
    {'isOverflow':0,'overflowParam':10,'deathMeterFlag':1,'deathMeterOne':15,'deathMeterTwo':30,'limitBuyFlag':1,'limitBuyMoney':1000,'isForceSrape':0,'forceSafeCheckYear':2,'forceScrapYear':12,'alarmValueOne':15,'alarmValueTwo':20}";
    * @return
    {"Result":"SUCCESS","CodeNumber":"20190008880003","SerialNo":1,"CMDHEXString":"68302019000888000304001048D3416444F1C12315691F2E5286FA9B8916"}
    */
    public static String writeSetParam(int Encrytype,String CodeNumber, int SerialNo, String jSON)

```

参数说明:

isOverflow: 过流使能
 overflowParam: 过流参数
 deathMeterFlag: 死表关阀使能
 deathMeterOne: 死表 1 关阀天数
 deathMeterTwo: 死表 2 关阀天数
 limitBuyFlag: 限购使能
 limitBuyMoney: 限购金额
 isForceSrape: 强制报废 0 报废;1 不报废;未使用
 forceSafeCheckYear: 设定安检时间
 forceScrapYear: 设定报废时间
 alarmValueOne: 报警金额 1
 alarmValueTwo: 报警金额 2

b. 回应(R)

```

    /**
     * 解析
     * @param Hexstring
     683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
     * @return {"CodeNumber":"20190008880007","SerialNo":1,"Data":
     {"Type":"MeterReplyWriteParam","Result":"SUCCESS"}}
     */
    public static String ParseData(int Encrytype,String Hexstring)

```

参数说明:

13.WriteGetRecords(读记录)

a. 下发(S)

```

/**
 * 读记录
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON ""
 */
public static String WriteGetRecords(int Encrytype,String CodeNumber, int
SerialNo, String json)

```

b. 回应(R)

```

/**
 * 解析
 * @param Hexstring
683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteGetRecords","Result":"SUCCESS","useGasAmountRecords":
{"useGasAmount1":0,"useGasAmount2":0,"useGasAmount3":0,"useGasAmount4":0,"useGas
Amount5":0,"useGasAmount6":0,"useGasAmount7":0,"useGasAmount8":0,"useGasAmount9"
:0,"useGasAmount10":0,"useGasAmount11":0,"useGasAmount12":0,"useGasAmount13":0,"
useGasAmount14":0,"useGasAmount15":0,"useGasAmount16":0,"useGasAmount17":0,"useG
asAmount18":0,"useGasAmount19":0,"useGasAmount20":0,"useGasAmount21":0,"useGasAm
ount22":0,"useGasAmount23":0,"useGasAmount24":0},"recordDate":"0000","safeCheckR
ecordsCount":0,"safeCheckRecords":{}}}}
 */
public static String ParseData(int Encrytype,String Hexstring)

```

参数说明：

useGasAmount1-24: 24 个月的累计用气量

recordDate: 记录报告时间

safeCheckRecordsCount: 安检记录条数（最多16）

safeCheckRecords: 安检记录内容

14.WriteGet24HoursRecords(读24小时记录)

a. 下发(S)

```

    /**
     * 读24小时记录
     *
     * @param Encrytype
     * @param CodeNumber
     * @param SerialNo
     * @param JSON    '{"startDate':'190821','readDays':2}";
     */
    public static String writeGet24HoursRecords(int Encrytype,String CodeNumber,
    int SerialNo, String jSON)
    参数说明:
        startDate: 起始日期
        readDays: 读取天数 (最多5天)

```

b. 回应(R)

```

    /**
     * 解析
     * @param Hexstring
    683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
     * @return{"CodeNumber":"20190008880007","SerialNo":1,"Data":
    {"Type":"MeterReplywriteGet24HoursRecords","Result":"SUCCESS","startDate":"19082
    1","readDays":0,"24hourUseGasData":{}}}
     */
    public static String ParseData(int Encrytype,String Hexstring)
    参数说明:

```

注意：当天记录不能读取。若下发命令天数超出记录范围，则回复天数为实际记录天数。

15.WriteGetModuleInfo(读模组信息)

a. 下发 (S)

```

    /**
     * 读模组信息
     *
     * @param Encrytype
     * @param CodeNumber
     * @param SerialNo
     * @param JSON    ""
     */
    public static String writeGetModuleInfo(int Encrytype,String CodeNumber, int
    SerialNo, String jSON)

```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring
683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteGetModuleInfo","Result":"SUCCESS","moduleType":"BC28JA-
02-
STD","moduleVersionType":"BC28JAR01A01_ONT","moduleIMEI":"867726031007158","card
IMSI":"460040944907093","cardICCID":"898602b9261771857093"}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:
moduleType: 模组型号
moduleVersionType: 模组版本号
moduleIMEI: 模组 IMEI
cardIMSI: 卡 IMSI
cardICCID: 卡 ICCID
```

16.WriteGetReportIPOne(读上报IP1)

a. 下发(S)

```
/**
 * 读上报IP1
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON ""
 */
public static String writeGetReportIPOne(int Encrytype,String CodeNumber,
int SerialNo, String jSON)
```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring
683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteGetReportIPOne","result":"SUCCESS","reportIP":"117.34.11
6.181","reportPort":"10001"}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:
```


17. WriteGetReportIPTwo(读上报IP2)

a. 下发(S)

```
/**
 * 读上报IP2
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON      ""
 */
public static String WriteGetReportIPTwo(int Encrytype, String CodeNumber,
int SerialNo, String jSON)
```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring
683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteGetReportIPOne","result":"SUCCESS","reportIP":"117.34.11
6.181","reportPort":"10001"}}
 */
public static String ParseData(int Encrytype, String Hexstring)
参数说明:
```

18. WriteGetReportIPThree(读上报IP3)

a. 下发(S)

```
/**
 * 读上报IP3
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON      ""
 */
public static String WriteGetReportIPThree(int Encrytype, String CodeNumber,
int serialNo, String jSON)
```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring
683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteGetReportIPOne","result":"SUCCESS","reportIP":"117.34.11
6.181","reportPort":"10001"}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:
```

19.WriteGetReportTimeOne(读上报时间1)

a. 下发(S)

```
/**
• * 读上报时间1
• *
• * @param Encrytype
• * @param CodeNumber
• * @param SerialNo
• * @param JSON ""
• */
• public static String WriteGetReportTimeOne(int Encrytype,String CodeNumber,
int SerialNo, String jSON)
```

b. 回应(R)

```
/**
• * 解析
• * @param Hexstring
683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
• * @return
{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteGetReportTimeOne","Result":"SUCCESS","reportType":1,"rep
ortTime":0,"reportSpace":1}}
• */
• public static String ParseData(int Encrytype,String Hexstring)
• 参数说明:
• reportType: 上报类型: 0 按小时上报 1 按日上报
• reportTime: 上报的起始时间
• reportSpace: 按小时上报时不超过 24 按日上报时不超过 28
```

20. WriteGetReportTimeTwo(读上报时间2)

a. 下发(S)

```
/**
 * 读上报时间2
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON ""
 */
public static String WriteGetReportTimeTwo(int Encrytype, String CodeNumber,
int SerialNo, String jSON)
```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring
683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return
{"Result": "SUCCESS", "CodeNumber": "20190008880007", "SerialNo": 1, "Data":
{"Type": "MeterReplyWriteGetReportTimeOne", "Result": "SUCCESS", "reportType": 1, "rep
ortTime": 0, "reportSpace": 1}}
 */
public static String ParseData(int Encrytype, String Hexstring)
参数说明:
```

21. WriteGetReportTimeThree(读上报时间3)

a. 下发(S)

```
/**
 * 读上报时间3
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON ""
 */
public static String WriteGetReportTimeThree(int Encrytype, String CodeNumber,
int SerialNo, String jSON)
```

b. 回应(R)

```

/****
* 解析
* @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
* @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteGetReportTimeOne","Result":"SUCCESS","reportType":1,"rep
ortTime":0,"reportSpace":1}}
*/
public static String ParseData(int Encrytype,String Hexstring)
参数说明:

```

22.WriteSetReportTimeOne(设置上报时间1)

a. 下发(S)

```

/****
* 设置上报时间1
*
* @param Encrytype
* @param CodeNumber
* @param SerialNo
* @param JSON    '{"reportType':0,'reportTime':1,'reportSpace':1}";
*/
public static String WriteSetReportTimeOne(int Encrytype,String CodeNumber, int
SerialNo, String jSON)
参数说明: (同读取)

```

b. 回应(R)

```

/****
* 解析
* @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
* @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteSetReportTimeOne","Result":"SUCCESS"}}
*/
public static String ParseData(int Encrytype,String Hexstring)
参数说明:

```

23.WriteSetReportTimeTwo(设置上报时间2)

a. 下发(S)

```

/****
* 设置上报时间2
*
* @param Encrytype
* @param CodeNumber
* @param SerialNo
* @param JSON    '{"reportType':0,'reportTime':1,'reportSpace':1}";
*/
public static String WriteSetReportTimeTwo(int Encrytype,String CodeNumber, int
SerialNo, String jSON)

```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
 {"Type":"MeterReplyWriteGetReportTimeOne","Result":"SUCCESS","reportType":1,"rep
 ortTime":0,"reportSpace":1}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:
```

24.WriteSetReportTimeThree(设置上报时间3)

a. 下发(S)

```
/**
 * 设置上报时间3
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON ""
 */
public static String WriteSetReportTimeThree(int Encrytype,String CodeNumber,
int SerialNo, String jSON)
```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
 {"Type":"MeterReplyWriteGetReportTimeThree","Result":"SUCCESS","reportType":1,"r
 eportTime":0,"reportSpace":1}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:
```

25.WriteSetReportIPOne(设置上报IP1)

a. 下发(S)

```

/**
 * 设置上报IP1
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON    '{"reportIP':'47.93.199.82','reportPort':6000}";
 */
public static String WriteSetReportIPOne(int Encrytype,String CodeNumber, int
SerialNo, String jSON)
参数说明:
reportIP:上报IP
reportPort:上报端口

```

b. 回应(R)

```

/**
 * 解析
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteSetReportIPTwo","Result":"SUCCESS"}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:

```

26.WriteSetReportIPTwo(设置上报IP2)

a. 下发(S)

```

/**
 * 设置上报IP2
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON    '{"reportIP':'47.93.199.82','reportPort':6000}";
 */
public static String WriteSetReportIPTwo(int Encrytype,String CodeNumber, int
SerialNo, String jSON)

```

b. 回应(R)

```

/**
 * 解析
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteSetReportIPTwo","Result":"SUCCESS"}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:

```

27. WriteSetReportIPThree(设置上报IP3)

a. 下发(S)

```
/**
 * 设置上报IP3
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON    '{"reportIP':'47.93.199.82','reportPort':6000}";
 */
public static String WriteSetReportIPThree(int Encrytype,String CodeNumber, int
SerialNo, String jSON)
```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterReplyWriteGetReportTimeOne","Result":"SUCCESS","reportType":1,"rep
ortTime":0,"reportSpace":1}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:
```

28. WriteSetAccumulatedGas(设置累积量)

a. 下发(S)

```
/**
 * 设置累积量
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON    '{"AccumulatedGas':'8888.88}";
 */
public static String WriteSetAccumulatedGas(int Encrytype,String CodeNumber, int
SerialNo, String jSON)
```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
 {"Type":"MeterReplyWriteGetReportTimeOne","Result":"SUCCESS","meterState":"11111
 1111111111"}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:
meterState:
```

29.WriteSetReSetMeter(清零)

a. 下发(S)

```
/**
 * 清零
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON ""
 */
public static String WriteSetReSetMeter(int Encrytype,String CodeNumber, int
SerialNo, String jSON)
```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
 {"Type":"MeterReplyWriteSetResetMeter","Result":"SUCCESS"}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:
```

30.WriteSetRepairMeter(维修)

a. 下发(S)

```
/**
 * 维修
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON ""
 */
public static String WriteSetRepairMeter(int Encrytype,String CodeNumber, int
SerialNo, String jSON)
```


b. 回应(R)

```
/**
 * 解析
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
 {"Type":"MeterReplywriteSetRepairMeter","Result":"SUCCESS"}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:
```

31.WriteSetSafeCheck(安检)

a. 下发(S)

```
/**
 * 安检
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON    '{"safeCheckJobNumber':'A00001'}";
 */
public static String WriteSetSafeCheck(int Encrytype,String CodeNumber, int
SerialNo, String jSON)
参数说明:
safeCheckJobNumber: 安检员编号 (固定长度)
```

b. 回应(R)

```
/**
 * 解析
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
 {"Type":"MeterReplywriteSetSafeCheck","Result":"SUCCESS"}}
 */
public static String ParseData(int Encrytype,String Hexstring)
参数说明:
```

32.WriteRequestMCUUpdate(升级请求)

a. 下发(S)

```
/**
 * 升级请求
 *
 * @param CodeNumber
 * @param SerialNo
 * @param JSON    "
{'versionCode':'20190820','totalFrameCount':105,'startAddress':'00004000','reg_c
rc_16':12345,'frameLength':992}";
 */
```

```
public static String writeRequestMCUUpdate(String CodeNumber, int SerialNo, String json)
```

参数说明: (以升级文件NB-FM-Meter(20190820).hex 为例 HEX文件格式详解:
<https://blog.csdn.net/a1037488611/article/details/43340055>)
versionCode:20190820
totalFrameCount:总帧数 (总数据长度/每帧长度 +(总数据长度%每帧长度>0?1:0))
startAddress:起始地址 (第一行的数据域+第二行地址域)
reg_crc_16:crc16_CCITT 固件校验码(校验码为所有数据检验码, 若最后一帧数据不足 每帧长度 需要以0xFF补齐) 该升级文件校验码为 A07D
frameLength:每帧长度

b. 回应(R)

```
/**
 * 不升级回应
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
 * {"Type":"MeterReplywriteRequestMCUUpdate","Result":"SUCCESS","ErrCodeNo":"'01'"}}
 */
```

```
public static String ParseData(String Hexstring)
```

参数说明:

01 地址超出范围。
10 表端有故障, 不能进行升级。
80 版本号错误(升级版本与表端当前版本一致)
XX 其它故障, 待定

```
/**
 * 升级回应
 * @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
 * @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
 * {"Type":"ResponsesMCUUpdateData","Result":"SUCCESS","versionCode":"20190820","frameIndex":13}}
 */
```

```
public static String ParseData(String Hexstring)
```

参数说明:

versionCode:版本
frameIndex:帧序号

说明: 如果表端可以升级, 则不回复“请求升级”命令, 直接发接 “请求数据帧”命令。

33.WriteMCUUpdateData(下发升级数据)

a. 下发(S)

```

/****
* 下发升级数据
*
* @param CodeNumber
* @param SerialNo
* @param JSON
{'versionCode':'20190820','frameLength':'496','resversionCode':'20190820','frame
Index':13,'totalFrameCount':215,'frameData':'5b1c934202d2e05c0028f9d1a869e618c01
aa861286ac01828622846fef766fd04e06a682078a968641c9047b442f8d32846fef771fd70bdfb
504460d4681b0243000902168880604d51022e0699143216000e00120a84201dd471b00e00027049
8a1697a191018081aa0612078c00602d42046fef73afd002608e003986268a168805d9047206a401
c761c206204988642f3db2078c0060ad52046fef726fd06e06268a16830209047206a401c2062384
67f1e0028f4dc07e000986268a168405d9047206a401c206228466d1e0028f3dc2046fef720fd207
8000602d5022005b0f0bd0120f7be711680268006a530501d5087009e0d30501d5088005e0130602d
5c21705c100e008600120704700007047f7fb50c461e468fb003ca089009914f007d0d00d16d1e002
108460020c0460321099a89050840002a03da8200801840080840800d01280a9002dd0020c0430a9
0089807430dd0f048f149281848430714189800281dd07542291e1ddd0a9842421be01898012802d
00020302104e00020f24305e02154401cb042fbdb30460022189b002121540f998b600a60486013b
0f0b0bdb1b6d1cdf70a9a002900da69420ba800f07afd0bab07cb03ab07c30899099800f0defd6b4
607c30146d5480918002d00910bddd24903980a9a49424018002303a90390684601f03bf808e0'}
* @return
{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"CMDHexString":"6
830201900088800070402007C2F31D4E3439F50D3BD0567E5DE9FBE30F550E2D0DF2C9CA5D1B07C2
82AA881C19C0C40E8411A9D99BF2D5FA27B03F2C85802C6962A950B9C3F228F93F91FF56EE540744
AF6F99BCF2A907515A5CAF29677AFD94AD76EB8035BE55FC4F14BED3E14089EEBDBBC086DD0451F5
83B74E84D2558B3C662298EEC2455F9A6A4129D1A9DC2C6CE9E258958DB5722B03E2ADBAAB2F9BC8
01E07B8E431E29F15287918520472979F5145898DF7255C2BEE35550D884C2288A31A398FF1DBC65
E42758099F09B1EC98D969385DE38120846BDA64F2591ADB53B2056A5A866969CC2452A3902CE7B5
7A28BFB839CDEDF4E311BB04010052C6FE4EFF6AAF400AE714151E1C8310E8931EAF2ABA3AD0A803
A56AEED5B53246EE3092E4C6D838B2812109EF28FCF79320771D0C30270FD539AB74182730D4F629
5EAB21A5787CEBE8A932BF243841C3D2CD5D058C29B739BCDBE8087914FE42DC79A8A6EC2DDC7E62
4A6817BCF62F9F0DCD6A3F7E47DCB6746F853376EE661014BA81140BA7A25D4CB6BF7A0800A96088
EF7B6BE2DCDAAD44B6877103F9BF0E084B7A6DD4F8465E13D6D3EACB49E307BBA082646A21D6882D
A3BAD895752472524DA0408E168772A66CAC923EF9F4E86225913B82FB75DF069605E1FD837A6879
4F7F909BF6D181739B6830E4724840BF6890947BAFFC3ACF2646579D66DDA7B1FE5D9AEF5FE5D4FF
3C024C34A16"}
*/
public static String writeMCUUpdateData(String CodeNumber, int SerialNo, String
JSON)

```

b. 回应(R)

```

/****
* 升级回应
* @param Hexstring
683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
* @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"ResponsesMCUUpdateData","Result":"SUCCESS","versionCode":"20190820","fr
ameIndex":13}}
*/
public static String ParseData(String Hexstring)

```

参数说明：

versionCode:版本

frameIndex:帧序号

/**

- 升级结束回应
- @param Hexstring
683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
- @return {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{ "Type":"WriteReportEndMCUUpdate","Result":"SUCCESS","versionCode":"20190820",
updateState:'00' }
*/
public static String ParseData(String Hexstring)

参数说明：

versionCode:版本

updateState:升级状态 00 成功 FF 失败

升级说明：

服务器在有升级需求时，先从升级文件中获取固件版本、起始地址、固件大小等信息，按设定的帧长度，计算总帧数，及校验数据。

服务器在收到表端的上报数据时通过“请求升级”命令发起升级事件，发送固件版本、起始地址、总帧数、每帧长度、校验数据等信息，表端准备好后，直接发送“请求数据帧”命令，接收升级数据。

所有数据接收完成后，表端通过“结束升级”命令发送数据校验结果给服务器。

关于续传：若升级过程中通讯超时或有其它故障时，升级中断，表端会保存已接收数据。当服务器下次接收到表端的上报数据时，应重新发送“请求升级”命令继续升级，表端根据已接收的帧序号继续发送“请求数据帧”命令接收数据

34.WriteDisConnect(断开)

a. 下发 (S)

```
/**
 * 断开
 *
 * @param Encrytype
 * @param CodeNumber
 * @param SerialNo
 * @param JSON    " ";
 */
public static String writeDisConnect(int Encrytype,String CodeNumber, int
SerialNo, String json)
```

参数说明：

特殊说明

- 1.返回值以 Result为准 SUCCESS成功 ERR错误
- 2.S为服务器下发解析 R为表端上报解析

35.ChecKSUMCode(检查校验和)

```

/****
* 检查校验和
*
* @param HexString :上报数据
*/
public static int CheckSUMCode(String HexString)
参数说明:
返回值: 0 校验成功
        1 校验失败

```

36.EncryptData_SE(加密机加密)

```

/****
* 调用py脚本加密数据（要求数据长度为16字节整数倍）
* @param SEUID :SEUID 协商密钥时存储。
* @param SESM2Key :SE 固定公钥（se初始化时写入）
* @param SETempSm2Key :SE 临时公钥
* @param Ranstr :随机数 （此处 可为空串“”）
* @param Msg :Msg 待加密/解密数据
* @return Msg :解密/加密后数据
*/
public static String EncryptData_SE(String SEUID, String SESM2Key, String
SETempSm2Key, String Ranstr, String Msg)
参数说明:

```

37.DecryptData_SE(加密机解密)

```

/****
* 调用py脚本加密数据（要求数据长度为16字节整数倍）
* @param SEUID :SEUID 协商密钥时存储。
* @param SESM2Key :SE 固定公钥（se初始化时写入）
* @param SETempSm2Key :SE 临时公钥
* @param Ranstr :随机数 （此处 可为空串“”）
* @param Msg :Msg 待加密/解密数据
* @return Msg :解密/加密后数据
*/
public static String DecryptData_SE(String SEUID, String SESM2Key, String
SETempSm2Key, String Ranstr, String Msg)
参数说明:

```

38.WriteSetEXAuthAPDU(外部认证)(传入参数为: SEUID,RandStr, 内部调用KMS接口)

b. 回应(R)

```

/****
* 解析
* @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
* @return{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterRequestExAuth","Result":"SUCCESS","SEUID":"8104515300000000","KeyT
ype":0,"RandStr": "10E561B4CF4DAD7D"}}
*/

```

```

public static String ParseData(int Encrytype,String Hexstring)

```

参数说明:

SEUID:SE UID

KeyType:更新卡密钥类型:0-测试密钥, 非0-正式密钥

RandStr:随机数

b. 下发(S)

```

/****
* 写入金额(充值)
* @param CodeNumber 条码号
* @param SerialNo 指令序号
* @param cmdParams SE相关参数 SEUID#KeyType#Randstr
* @return
{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"CMDHEXString":"6
83020190008880007040020D0100184320210080000300C40D5038E80000000000000000000000
00000009F16"}
*
*
public static String WriteSetEXAuthAPDU(int Encrytype,String CodeNumber,
int SerialNo, String cmdParams)

```

39.WriteGetUpdateCardKeyADPU(更新SE卡密钥)(传入参数为: SEUID,RandStr, 内部调用KMS接口)

b. 回应(R)

```

/****
* 解析
* @param Hexstring 683020190008880007840010b820c4123e70436bef91a5ff2fc2b0e9b416
* @return{"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"Data":
{"Type":"MeterRequestUpdateCardKey","Result":"SUCCESS","SEUID":"8104515300000000
","KeyType":0,"RandStr": "10E561B4CF4DAD7D"}}
*/

```

```

public static String ParseData(int Encrytype,String Hexstring)

```

参数说明:

SEUID:SE UID

KeyType:更新卡密钥类型:0-测试密钥, 非0-正式密钥

RandStr:随机数

b. 下发(S)

```

    /**
    * 写购入金额(充值)
    * @param CodeNumber 条码号
    * @param SerialNo 指令序号
    * @param cmdParams SE相关参数 SEUID#KeyType#Randstr
    * @return
    {"Result":"SUCCESS","CodeNumber":"20190008880007","SerialNo":1,"CMDHEXString":"6
    83020190008880007040020D0100184320210080000300C40D5038E80000000000000000000000
    00000009F16"}
    */
    public static String WriteGetUpdateCardKeyADPU(int Encrytype,String
    CodeNumber, int SerialNo, String cmdParams)

```

40.WriteGetMeterPulseInfo(读取脉冲数据)

a. 下发(S)

```

    /**
    * 读取脉冲数据
    *
    * @param Encrytype
    * @param CodeNumber
    * @param SerialNo
    * @param JSON '{"PulseDay':0}"; 0: 昨天数据 1: 前天数据 2: 大前天数据。。。
    */
    public static String WriteGetMeterPulseInfo(int Encrytype,String CodeNumber, int
    SerialNo, String json)

```

b. 回应(R)

```

/****
* 解析
* @param Hexstring
* @return{"Result":"SUCCESS","CodeNumber":"20200000000001","SerialNo":1,"Data":
{"Type":"MeterReportPulseInfo","Result":"SUCCESS","FinalFlag":1,"pulseDate":"200
810","pulseData":
[{"startTime":"00:00","PulseData":"1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2
|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10
|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|
9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|
7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|
5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|
3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|
1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9
|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7
|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5
|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3
|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1
|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|10|1|2|3|4|5|6|7|8|9|
10|"}]}}
*/

```

public static String ParseData(int Encrytype,String Hexstring)

参数说明:

FinalFlag: 脉冲数据是否发送完成标识 **0** 最后一帧 其他值: 还有数据, 需要继续 发送获取脉冲数据的指令

pulseDate: 脉冲数据所属日期

StartTime: 脉冲段起始时间

PulseData: 脉冲个数

上述数据表明: **00:00** 1个脉冲, **00: 01** 2个脉冲 **00: 03** 三个脉冲。。。以此类推

特殊说明

- 1.返回值以 Result为准 SUCCESS成功 ERR错误
- 2.S为服务器下发解析 R为表端上报解析
- 3.py脚本加解密说明:加解密使用密钥均为协商密钥指令写入到se中, 使用密钥标识为0702 加密密钥, 0902 解密密钥。因KMS中不存储协商密钥, 所以每次调用kms解密/加密数据时, 需要传入SEkey,SETempKey, SEUID 以用于KMS计算协商密钥后 进行加/解密

交互流程

1. 每次交互都由表端上报数据开始, 若表端上报数据后 30 秒内没有收到服务器的数据, 则重新上报, 最多重试 3 次。
2. 服务器收到表端的上报数据后, 若上报数据有误则不响应, 上报数据无误时, 如有下发指令, 则 发送指令, 没有指令则发送“结束通讯”指令。
3. 表端收到服务器的指令后, 根据指令做出回复。如为“结束通讯”指令则立即断开网络连接, 结束交互。表端回复后再次接收服务器指令, 如 30 秒内没有接收到指令则结束交互。