易连充电宝协议接入手册

版本：V1.1

更新日期：2025 年 12 月 27 日

本产品的应用说明书如有变更，恕不另行通知。

深圳市易连物联网有限公司保留在不另行通知的情况下，对其中所包含的材料进行更改的权利，同时由于信任所引用的材料所造成的损害（包括结果性损害），包括但不限于印刷上的错误和其他与此出版物相关的错误，易连物联网将不承担责任。

修改记录

文档版本	撰写者	测试者	审核者	发布日期	修改说明
V1.0	Lz	/	Lxl	2025/12/24	初版
v1.1	Lz	/	/	2025/12/27	1、 增加 0x20 获取设置充电宝健康充电次数指令 2、 增加握手加密流程图

目录

1 概述

1.1 本文档适用于客户自己开发 BLE 免费接入易连电宝管家小程序。

1.2 文档会保持更新，以官网链接为最新版本。

2 说明

2.1 充电宝 CID 固定默认为 0x008B，VID 固定默认为 0x00CB，PID 默认为 0x0001，使用此 CID PID VID 接入本公司电宝管家小程序时，页面为本公司公版，如需个性化差异化定制，需向本公司申请 PID，请联系客户经理：邹曼娜 19065037169。申请定制化 PID 后，提供新的密钥，加密方法 1 使用对应新的密钥。

2.2 电宝管家小程序二维码：

3 产品及流程

3.1 产品工作流程

1. BLE 上电开机。
2. 打开电宝管家小程序，搜索设备，连接设备。
3. 小程序连接上 BLE 后马上双向握手交互。
4. BLE 连接上电宝管家小程序后进行数据通信。

4 蓝牙接口（默认）

4.1 蓝牙名称：AiLink_xxxx

注：xxxx 为 Mac 地址后 4 个字符，通过扫描应答包广播出去

4.2 广播数据

使用我司充电宝小程序接入的设备，需根据我司要求的格式进行设置。
广播类型(AD_TYPE_FLAGS) ： 0x06
广播 UUID ： UUID1 = 0xFFE0 ,UUID2= 0xFEE0 ；

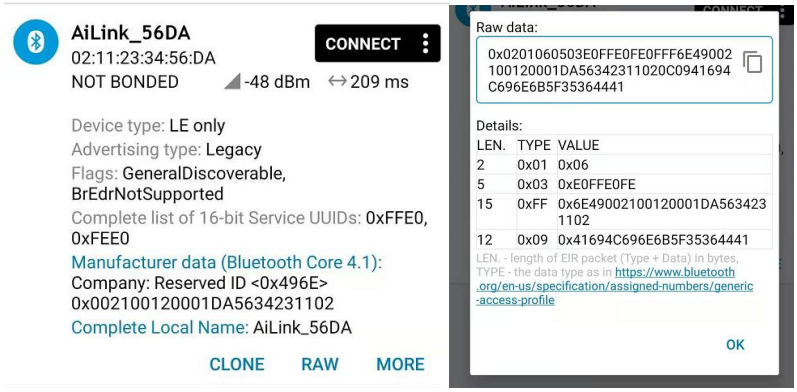
广播自定义数据（0xFF）：
设备自定义广播数据内容包含

1、Company ID 。固定：0x6e49
2、CID：产品类型（2bytes）CID；
3、VID：厂商 ID （2byte） VID
4、PID：产品 ID （2byte） PID
5、Mac 地址（MAC 是固定的，小端序）

例如广播出来的数据为：

6e49002F0003000112613423110532201400 2000

6e49：为 In，
002F 是 CID，表示产品类型，
0003 是 VID，表示厂商 ID，
0001 是 PID，表示产品 ID。
126134231105 是 Mac 地址，因为是小端序，所以 Mac 地址是：02 ： 11 ： 23 ： 34 ： 61 ： 12
蓝牙工具显示如下图：



4.3 UUID 说明

易连物联网的 APP 交互使用的服务 UUID 为 FFE0。

4.4 蓝牙连接服务列表：FFE0

4.4.1 服务 UUID：

0000FFE0-0000-1000-8000-00805F9B34FB

4.4.2 特征值 UUID1：

0000FFE1-0000-1000-8000-00805F9B34FB

属性：read,write,write no response

功能 ：APP 下发的指令会通过此 UUID 传输给 BLE 设备

4.4.3 特征值 UUID2：

0000FFE2-0000-1000-8000-00805F9B34FB

属性：read,notify

功能： BLE 设备上报的数据会通过此 UUID 传输给 APP

4.4.4 特征值 UUID3：

0000FFE3-0000-1000-8000-00805F9B34FB

属性：read,write,write no response,notify

功能：APP 与 BLE 进行握手交互和设备通用指令传输的 UUID，有 write 和 notify

5 交互指令数据格式

5.1 通用指令格式

Byte	Value	Description
0	0xA6	包头
1		Payload 长度（最大 16byte）
2 ~n		Payload
n+1	SUM（1~n）	(1~n)校验和
n+2	0x6A	包尾（注：n+2 不能超过 20）byte1 + byte2 + …+byte n 的和，取低位 1 byte。

设置指令里，数据的 Byte 数不能超过 20

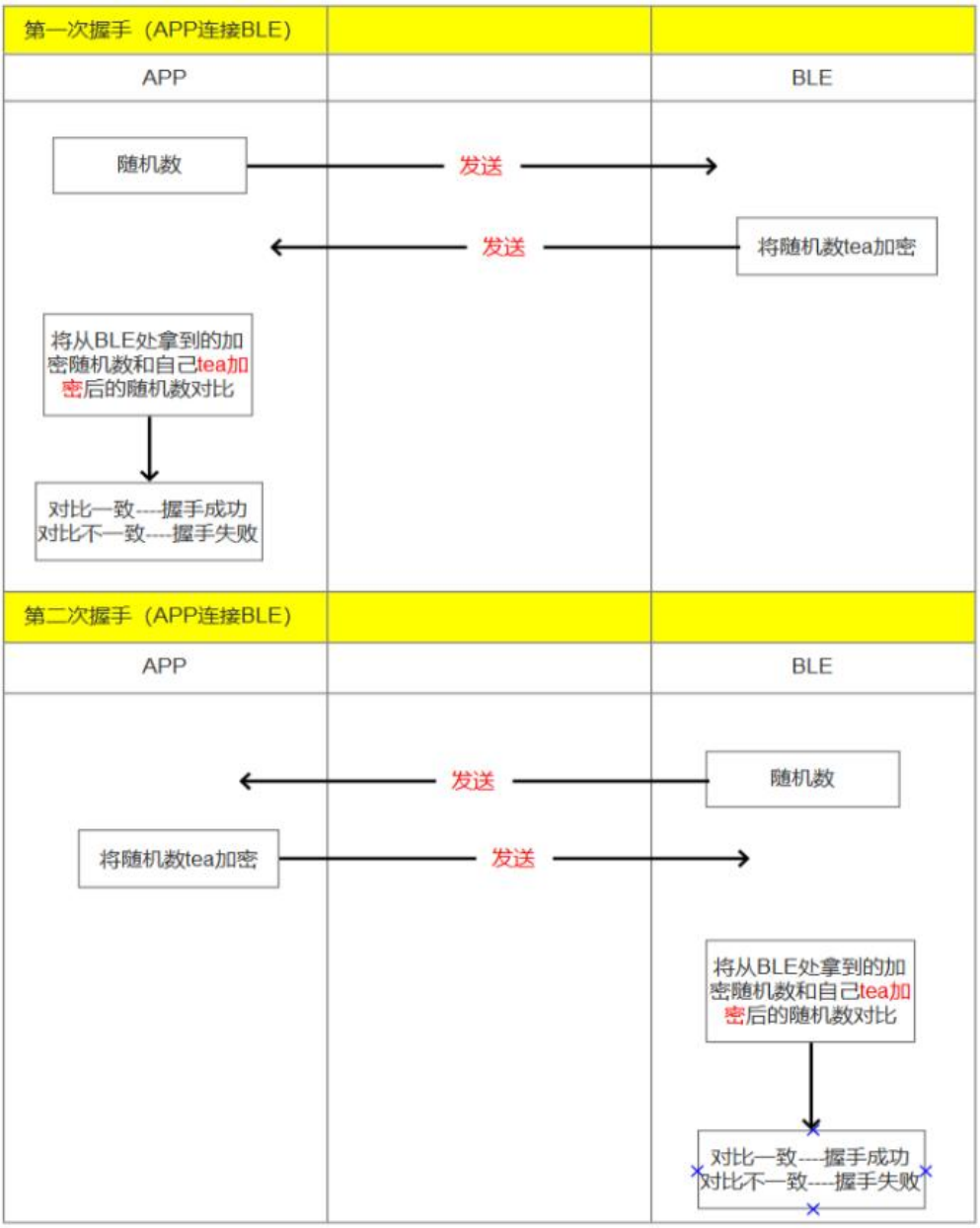
5.2 充电宝产品指令格式

Byte	Value	Description	
0	0xA7	包头	
1	0x00	产品类型（CID）高字节	
2	0x8B	产品类型（CID）低字节	
3		Payload 长度(payload 部分的字节数量)	
4		OP	Payload
5		T	
6		L	
7		V	
....		...	
n+1	SUM（1~n）	（1~n）校验和(累加和,取低八位)	
n+2	0x7A	包尾	

检验和是指 byte1 + byte2 + … +byte n 的和，取低位 1 byte。

6 双向握手交互（模块和 app 连接时需要握手）

6.1 握手流程



握手流程（通过 UUID3）

1. APP 校验 BLE 设备（第 1 次握手）
   (1) APP 连接后，向 BLE 设备发起握手指令 0x23，握手指令里含一组随机数。
   (2) BLE 设备收到握手指令后，将随机数加密后通过指令 0x24 返回。
   (3) APP 收到返回后的数据，校验是否正确。
   (4) 校验出错则断开 BLE 设备。
   (5) 若超时 5s 收不到 BLE 设备返回的数据，也断开连接。

2. BLE 设备校验 APP（第 2 次握手）
   (1) BLE 设备在 APP 发起握手后，将随机数加密返回后，即第 1 次握手完成后。
   (2) BLE 设备再主动发起握手校验，握手指令 0x23 里含一组随机数。
   (3) APP 收到 BLE 设备发起的握手校验，将得到的随机数加密通过指令 0x24 返回。
   (4) BLE 设备收到数据后，校验是否正确。
   (5) 校验出错则断开 APP 连接。
   (6) 若超时 5s 收不到 APP 返回的数据，也断开连接。

6.2 APP 设置握手信息（0x23）

指令格式：

Byte	Value	Description	
0	0xA6	包头	
1	0x11	Payload 长度，17Byte	
2	0x23	功能字：	Payload
3-18	Source data	16byte 源数据 APP 下发加密前的明文数据，BLE 设备将会根据此数据采用加密 1 方法加密数据，	
19		校验和(1~18)	

6.3 BLE 设备返回加密后的握手信息（0x24）

指令格式：

Byte	Value	Description	
0	0xA6	包头	
1	0x11	Payload 长度，17Byte	
2	0x24	功能字：	Payload
3-18	Encrypted data	16byte 加密数据 BLE 设备根据 APP 下发的加密前的明文数据，采用加密 1 方法得到此加密数据	
19		校验和(1~18)由加密后的数据来计算校验和	

6.4 BLE 设备设置握手信息（0x23）

指令格式：

Byte	Value	Description
0	0xA6	包头

1	0x11	Payload 长度，17Byte	
2	0x23	功能字：	Payload
3-18	Source data	16byte 源数据 BLE 设备上传加密前的明文数据，APP 将会根据此数据采用加密 1 方法加密数据	
19		校验和(1~18)	

6.5 APP 返回加密后的握手信息（0x24）

指令格式：

Byte	Value	Description	
0	0xA6	包头	
1	0x11	Payload 长度，17Byte	
2	0x24	功能字：	Payload
3-18	Encrypted data	16byte 加密数据 APP 根据 BLE 设备上传的加密前的明文数据，采用加密 1 方法得到此加密数据	
19		校验和(1~18)由加密后的数据来计算校验和	

7 充电宝产品指令集

- 数据格式：

1. 内容说明：

OP:操作指令,详情看列表

TLV:

T:type,数据类型(1 字节)

L:length,数据长度,即 V 的长度(1 字节)

V:value,数据内容(N 字节)

一条指令里可以拼接多个 TLV.

Byte	Value	Description
0	0xA7	包头
1	0x00	产品类型（CID）高字节
2	0x8B	产品类型（CID）低字节
3		Payload 长度(payload 部分的字节数量)

4		OP	Payload
5		T	
6		L	
7		V	
....		...	
n+1	SUM（1~n）	（1~n）校验和(累加和,取低八位)	
n+2	0x7A	包尾	

检验和是指 byte1 + byte2 + … +byte n 的和，取低位 1 byte。

7.1 OP 列表

op 值	功能描述	备注
0x01	App 设置功能	
0x02	App 查询信息	app 连接后需要发指令查询设备支持的功能信息列表
0x03	设备主动返回状态信息	设备实时读取充电宝状态，主动返回当前状态数据给 APP 例：充电宝正在充电或者放电时，设备返回 1、电池（NTC 获取）温度 2、主板（芯片内置温度传感器）温度 3、电量 4、电流 5、电压 6、功率 7、充电时间 8、充电量 9、充电 1%电量的时间（充电时才会返回此信息） 10、充满电还需时间（充电时才会返回此信息） 设备工作时，充电宝无状态时，设备返回 1、电池（NTC 获取）温度 2、主板（芯片内置温度传感器）温度 3、电量 设备边充边放时，设备返回 1、电池（NTC 获取）温度 2、主板（芯片内置温度传感器）温度 3、电量 4、电流（总） 5、电压（总）

		6、功率（总） 7、放电时间 8、充电时间

7.2 TLV 列表

如果设备不支持 type，则设备返回 length=1 ， value=0xFF

Type	Length	Value uint8 无符号一个字节 ,int8 有符号一个字节 （如果没有特别说）超过一个字节的数据使用 小端序发送
0x01 设备功能列表 （只查）	N	根据下面充电宝功能列表决定字节数 uint8：第一个字节 uint8：第二个字节 ....
0x02 充电宝充电状态（只查）	1	uint8： 0：充电宝正在充电 1：充电宝没有在充电

0x03 充电宝放电状态和电流电压值/充电宝放电的端口（只查）	N	uint8 ：充电宝正在放电的端口数量 0~255 0 表示充电宝没有端口在放电
		{
		端口 1：
		uint8 ： 端口的类型 （0：Type-C 口 1：Type-A 口 ...）
		uint8 ： 端口的编号 0-255
		uint16： 放电时间 单位 s 分度 1s
		uint8： 放电量 单位% 分度 1% 如果设备不支持获得单个端口的放电量 则 uint8=0xFF
		uint24： 电流值 单位 mA 分度 0.01mA
		uint24： 电压值 单位 mV 分度 0.01mV
		如果设备不支持获得单个端口的电流电压值 则这 uint24=0xFF FF FF。
		}
		
		{
		端口 2 （如果有多个端口数量处于放电状态）
		uint8 ： 端口类型
		uint8 ： 端口编号
		uint16： 放电时间 单位 s 分度 1s
		uint8： 放电量 单位% 分度 1% 如果设备不支持获得单个端口的放电量 则 uint8=0xFF

		uint24： 电流值 单位 mA 分度 0.01mA
		uint24： 电压值 单位 mV 分度 0.01mV
		如果设备不支持获得单个端口的电流电压值 则这 uint24=0xFF FF FF。
		}
		
		{
		端口 3 同上格式
		}

0x04 当前电量 （只查）	1	uint8 ： 电量的大小 单位% 0~100
0x05 充电宝充电总电流（只查）	3	uint24： 单位 mA 分度 0.1mA
0x06 充电宝放电总电流（只查）	3	uint24： 单位 mA 分度 0.1mA
0x07 充电宝充电总电压（只查）	3	uint24： 单位 mV 分度 0.1mV
0x08 充电宝放电总电压（只查）	3	uint24： 单位 mV 分度 0.1mV
0x09 VBAT/电池 电压（只查）	3	uint24： 单位 mV 分度 0.1mV
0x0A 电池（NTC 获取）温度（只查）	2	int16：单位° C 分度 0.1° C
0x0B 主板(芯片内置温度传感器）温度（只查）	2	int16：单位° C 分度 0.1° C
0x0C 当前电池容量 （只查）	3	uint24： 单位 mAH 分度 1mAH
0x0D 充电宝充电功率(只查）	3	uint24： 单位 W 分度 0.01W
0x0E 充电宝放电功率(只查）	3	uint24： 单位 W 分度 0.01W
0x0F PD 快充功率（只查）	3	uint24： 单位 W 分度 0.01W
0x10 标配充电器功率（只查）	3	uint24： 单位 W 分度 0.01W
0x11 无线充电功率（只查）	3	uint24： 单位 W 分度 0.01W
0x12 充电宝充电次数(只查）	2	uint16：次数的大小
0x13 出厂时间 （查/设）	3	uint8 ： yy 年份 2000+yy=实际年份 uitn8 ： 月份 uitn8 ： 日
0x14 激活时间 （查/设）	6	uitn8 ： yy 年份 2000+yy=实际年份

		uitn8 ： 月份 uitn8 ： 日 uitn8 ： 时 uitn8 ： 分 uitn8 ： 秒
0x15 关闭打开全部端口快充功能（设）	1	uitn8 ： 0-打开 1-关闭 （app 设置 设备不写） uitn8 ： 0-成功 1-失败 （设备回应 app 不写）
0x16 获得充电宝充电时间 （查）	2	uint16：单位 s 分度 1s

0x17 获得充电宝的端口的放电时间（查）	N	{
		端口 1：
		uint8 ： 端口的类型 （0：Type-C 口 1：Type-A 口 ...）
		uint8 ： 端口的编号 0-255
		uint16：单位 s 分度 1s
		}
		{
		端口 2：
		uint8 ： 端口的类型 （0：Type-C 口 1：Type-A 口 ...）
		uint8 ： 端口的编号 0-255
		uint16：单位 s 分度 1s
		}
		{
		端口 3：格式同上
		}

0x18 获得充电宝总充电量（查）	1	uint8：单位% 分度 1% 当充电宝边充边放时，此时的充放电量为净电量，不是实际充放电量
0x19 获得充电宝总放电量（查）	1	uint8：单位% 分度 1%
0x1A 出厂时设置的电池容量（查/设）	3	uint24：单位 mAH 分度 1mAH
0x1B 获得设备端口数量（查）	1	uint8：单位 个 分度 1 个
0x1C 设置获取设备 SN 号（查/设）	N	N SN 号的 ASCLL 值
0x1D 获取充电宝充电 1%电量所需时间（查）	2	uint16：单位 s 分度 1s
0x1E 获取充满电还需时间（查）	2	uint16：单位 s 分度 1s
0x1F 获取每节电池的电压（查）	N	uint8 ： 电池的节数 1-8 最多 8 节电池 { 电池 1： uint8：电池编号 1 uint24： 单位 mV 分度 0.1mV

		}
		{
		电池 2：
		uint8：电池编号 2
		uint24： 单位 mV 分度 0.1mV
		}
		{电池 3：格式同上}

0x20 获取设置充电宝健康充电次数（查/设）	2	uint16： 单位次 分度 1 次

- APP 为了保证 APP 的功能和设备端的功能同步,APP 连接设备时,需要获取设备支持的功能.
- 若设备不支持功能,则相对应的功能和指令则不需支持及设计.

充电宝功能列表(0 不支持,1 支持)

Value	信息
Byte 1 Bit0	快充指示
Byte 1 Bit1	充电宝充电和放电状态指示
Byte 1 Bit2	端口状态指示
Byte 1 Bit3	输入输出总 VOUT 电压（输出给设备的电压）
Byte 1 Bit4	输入输出总 IBUS 电流
Byte 1 Bit5	VBAT 电压（电池电压）
Byte 1 Bit6	充电宝温度
Byte 1 Bit7	芯片温度
Byte 2 Bit0	当前电池容量
Byte 2 Bit1	电量
Byte 2 Bit2	充电宝充电和放电的功率
Byte 2 Bit3	电池端充放电功率
Byte 2 Bit4	保持设备工作
Byte 2 Bit5	关闭打开快充功能
Byte 2 Bit6	Pd 快充功率
Byte 2 Bit7	无线充电功率
Byte 3 Bit0	每个端口的单独输出电流
Byte 3 Bit1	每个端口的单独输出电压
Byte 3 Bit2	每个端口的单独充电电量
Byte 3 Bit3	每个端口的单独放电电量
Byte 3 Bit4	获取单节电池的电压

说明：支持功能列表持续更新，功能字节数随功能数量变化

APP 设置：

Byte	Default	Description

0	0xA7	包头	
1~2	0x008B	产品类型：充电宝	
3		Payload 长度	
4	0x01	CMD：设置	Payload
	TYPE		
	Length		
	Value		
	TYPE		
	Length		
	Value		
	...	...	
	SUM	校验和	
	0x7A	包尾	

APP 查询：

Byte	Default	Description	
0	0xA7	包头	
1~2	0x008B	产品类型：	
3		Payload 长度	
4	0x02	CMD：设置	Payload
	TYPE1		
	TYPE2		
	...	...	
	SUM	校验和	
	0x7A	包尾	

MCU 返回/主动返回信息:

Byte	Default	Description	
0	0xA7	包头	
1~2	0x008B	产品类型：	
3		Payload 长度	
4	CMD	CMD：MCU 返回参数功能	Payload
	TYPE		
	Length		
	Value		
	TYPE		
	Length		
	Value		
	...	...	
	SUM	校验和	
	0x7A	包尾	

8 BLE 通用指令集

指令格式

Byte	Value	Description
0	0xA6	包头
1		Payload 长度（最大 16byte）
2 ~n		Payload
n+1	SUM（1~n）	(1~n)校验和
n+2	0x6A	包尾（注：n+2 不能超过 20）byte1 + byte2 + …+byte n 的和，取低位 1 byte。

设置指令里，数据的 Byte 数不能超过 20

8.1 设置蓝牙名称 Type = 0x01

App 发送：

Byte	Value	Description	
0	0xA6	包头	
1	Len	Payload 长度（最大 16byte）	
2	0x01	Type：设置蓝牙名称	Payload
3 ~ n	Name	名称（需要对应 ASCII 表）	
n+1	Num	MAC 字符个数：名称后面跟随的 MAC 字符的个数 0x00：代表没有，则是固定蓝牙名称。 0x01：代表后面带有 mac 地址的 1 个字符，例如： Swan_x。 0x02：代表后面带有 mac 地址的 2 个字符，例如： Swan_xx。 默认 Num=4；Num 最大为 12 注：Name 长度+“_”+Num 最大为 15	
n + 2	Sum（1~n）	校验和	
N+3	0x6A	包尾	

设备响应：

Byte	Value	Description	
0	0xA6	包头	
1	0x02	Payload 长度	
2	0x01	Type：回复设置蓝牙名称结果	Payload

3		结果值： 0x00：成功（立即生效） 0x01：失败 0x02：不支持	
4	Sum（1~3）	校验和	
5	0x6A	包尾	

设置蓝牙名称可以设置为固定字符作为蓝牙名称，例如设置为 swan，所有的模块都会显示为 swan。同时也可以设置为固定蓝牙名称+“_”+Mac 地址的方式，这样子有利于每个模块的名称都有差异。

- 举例 ： 蓝牙的 MAC 地址为 12 : 34 : 56 : 78 : 9A : BC。

8.2 读取 BLE 版本号（Type：0x46）

App 发送：

Byte	Value	Description	
0	0xA6	包头	
1	0x01	Payload 长度	
2	0x46	Type：读取 BM 模块软硬件版本号	Payload
3	0x0F	(1~2)校验和	
4	0x6A	包尾	

设备响应：

Byte	Value	Description	
0	0xA6	包头	
1		Payload 长度	
2	0x46	Type：回复 BM 模块软硬件版本号	Payload
3		针对于某些芯片 MTU 不支持高于 23bytes,一条指令发不完版本号,固可分包发送. Bit7-4:指令条数(当只有 1 条指令时,该值为 0,类推) Bit3-0:当前指令 index(当是第一条指令时,该值为 0,类推)	
4-N		版本号(ASCII 字符) 例如:LM09BH1S1.0.0_01200105	
N+1	Sum	校验和	
N+2	0x6A	包尾	

9 加密方式

9.1 加密方式 1（双向握手加密）

加密方式： TEA 加密 （对 byte3~18 的数据进行加密）适用于通用指令集 A6 指令
加密秘钥： 充电宝默认密钥：const uint32_t tea_key[4] = {0x9fe98d00, 0x0e79553e, 0xb867e466, 0x25c95564}；申请定制化 PID 后，提供新的密钥，加密方法 1 使用对应新的密钥。
加密函数： encrypt_8byte(data,tea_key);//这里的数据 data 是 byte3 开始的指针

```
void encrypt_tea(uint32_t *v, uint32_t *k) // 8byte 数据加密
{
    uint32_t v0 = v[0], v1 = v[1], sum = 0, i;                  /* set up */
    uint32_t delta = 0x9e3779b9;                                /* a key schedule constant */
    uint32_t k0 = k[0], k1 = k[1], k2 = k[2], k3 = k[3]; /* cache key */
    for (i = 0; i < 32; i++)
    { /* basic cycle start */
        sum += delta;
        v0 += ((v1 << 4) + k0) ^ (v1 + sum) ^ ((v1 >> 5) + k1);
        v1 += ((v0 << 4) + k2) ^ (v0 + sum) ^ ((v0 >> 5) + k3);
    } /* end cycle */
    v[0] = v0;
    v[1] = v1;
}
// 加密函数
void ailink_encrypt(uint8_t *p) // 16byte 数据加密
{
    union u8_to_u32
    {
        uint8_t u8_data[16];
        uint32_t u32_data[4];
    } v;
    memcpy((uint8_t *)&v.u8_data, p, sizeof(v));
    encrypt_tea(&v.u32_data[0], (uint32_t *)tea_key);
    encrypt_tea(&v.u32_data[2], (uint32_t *)tea_key);
    memcpy(p, (uint8_t *)&v.u8_data, sizeof(v));
}
// 8 byte 加密函数
void encrypt_8byte(uint8_t *p, uint32_t *tea_key1) // 8byte 数据加密
{
    union u8_to_u32
    {
        uint8_t u8_data[8];
```

```
        uint32_t u32_data[2];
    } v;
    memcpy((uint8_t *)&v.u8_data, p, sizeof(v));
    encrypt_tea(&v.u32_data[0], (uint32_t *)tea_key1);
    memcpy(p, (uint8_t *)&v.u8_data, sizeof(v));
}
```

9.2 A7 指令 Payload 加密方式

充电宝 CID 固定默认为 0x008B，VID 固定默认为 0x00CB，PID 默认为 0x0001。

```
typedef struct{
    uint8_t mac[6];//mac 地址
    uint8_t productType[2];//cid
    uint8_t *data;//要加密的数据
    uint8_t len;//数据长度
}encryptMacAndProduct_t;      //A7 数据加密使用

void api_encryptMacAndProductType(encryptMacAndProduct_t *encryData)
{
    if (!encryData->productType[0])
    {
        encryData->productType[0]++;
    }
    if (!encryData->productType[1])
    {
        encryData->productType[1]++;
    }
    for (uint8_t i = 0; i < encryData->len; i++)
    {
        encryData->data[i] ^= encryData->mac[i % 6];
        encryData->data[i] ^= encryData->productType[i % 2];
    }
}
static void encry_a7_payload(uint8_t *data, int len)
{
#if 1 // A7  指令加密  解密
    uint8_t i, checkSum = 0;
    encryptMacAndProduct_t encryptData;

    memcpy(encryptData.mac, &elink_gb_data.mac[0], 6);
    encryptData.productType[0] = elink_gb_data.cvpid[0];
    encryptData.productType[1] = elink_gb_data.cvpid[1];
```

```
    encryptData.data = data;
    encryptData.len = len;
    api_encryptMacAndProductType(&encryptData);
    // 加密时有效，解密时无作用
    checkSum = 0;
    checkSum += elink_gb_data.cvpid[0];
    checkSum += elink_gb_data.cvpid[1];
    checkSum += len;
    for (i = 0; i < len; i++)
    {
        checkSum += data[i];
    }
    data[len] = checkSum;
#endif
}
```

10 联系我们

深圳市易连物联网有限公司

地址：深圳市宝安区西乡街道银田工业区侨鸿盛文化创意园写字楼 A 栋五层 502 室

Tel: +（86）0755-81773367

市场部邮箱：marketing@elinkthings.com

FAE 邮箱：hw@elinkthings.com

官网： www.elinkthings.com