**1、登录**

**桩发送：**68 01 00 00 00 1D 10 00 02 54 84 56 18 35 02 02 00 00 00 00 00 00 00 01 00 01 02 00 00 00 04 00 00 00 01 2B D9

解析：

68 -- 起始标识

01 -- 命令码

00 00 -- 消息流水号，当前为0

00 1D -- 消息体长度，值为29

10 00 02 54 84 56 18 35 桩编号，此示例为1000025484561835

02 -- 桩类型，值为2，表示直流充电桩

02 -- 充电枪数量，值为2，表示双枪

00 00 00 00 -- 运营商编码

00 00 00 -- 密码

01 00 -- 桩软件版本号，当前为V1.0

01 02 -- 通信协议，当前为V1.2

00 00 00 04 -- 计费规则ID，当前为4

00 00 00 01 -- 计费规则版本号，当前为1

2B D9 -- CRC校验

**后台回复：**68 81 00 00 00 09 10 00 02 54 84 56 18 35 00 A5 EF

解析：

68 -- 起始标识

81 -- 命令码

00 00 -- 消息流水号，响应端的消息流水号保持与请求时的流

水号一致，桩发送为0，所以这里也为0

00 09 -- 消息体长度，值为9

10 00 02 54 84 56 18 35 -- 桩编号

00 -- 结果，0表示登录成功

A5 EF -- CRC校验

**2、计费规则设置**

**后台发送：**68 0B 00 01 00 22 00 00 00 04 00 00 00 01 17 07 04 10 59 09 13 88 02 58 07 D0 05 14 05 14 05 14 05 14 01 00 00 23 59 03 65 93

解析：

68 -- 起始标识

0B -- 命令码

00 01 -- 消息流水号

00 22 -- 消息体长度

00 00 00 04 -- 计费规则ID

00 00 00 01 -- 计费规则版本号

17 07 04 10 59 09 -- 生效时间，代表2017-07-04 10:59:09

13 88 -- 预约费单价，1388H = 5000D，表示5元

02 58 -- 服务费单价，0258H = 600D，表示0.6元

07 D0 -- 停车费单价，07D0H = 2000D，表示2元

05 14 -- 尖时电价，0514H = 1300D, 表示1.3元

05 14 -- 峰时电价，0514H = 1300D, 表示1.3元

05 14 -- 平时电价，0514H = 1300D, 表示1.3元

05 14 -- 谷时电价，0514H = 1300D, 表示1.3元

01 -- 时段数，当前值为1，表示只有一个时段

00 00 -- 时段开始时间00:00

23 59 -- 时段结束时间23:59

03 -- 时段类型，值为3，表示时段为平时段

65 93 -- CRC校验

**桩回复：**68 8B 00 01 00 09 00 00 00 04 00 00 00 01 00 22 5D

解析：

68 -- 起始标识

8B -- 命令码

00 01 -- 消息流水号，响应端的消息流水号保持与请求时的流

水号一致，后台发送为1，所以这里也为1

00 09 -- 消息体长度

00 00 00 04 -- 计费规则ID

00 00 00 01 -- 计费规则版本号

00 -- 结果，0表示成功

22 5D -- CRC校验

**3、心跳**

**桩发送：**68 0C 00 03 00 03 02 01 01 09 FB

解析：

68 -- 起始标识

0C -- 命令码

00 03 -- 消息流水号

00 03 -- 消息体长度

02 -- 枪数量

01 -- A枪状态

01 -- B枪状态

09 FB -- CRC校验

**后台回复：**68 8C 00 03 00 06 17 07 04 10 59 29 CB ED

解析：

68 -- 起始标识

8C -- 命令码

00 03 -- 消息流水号

00 06 -- 消息体长度

17 07 04 10 59 29 -- 服务器时间

CB ED -- CRC校验

**4、CRC校验代码(JAVA版)**

package boss.server.utils;

import org.apache.commons.codec.DecoderException;

import org.apache.commons.codec.binary.Hex;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* CRC校验工具类

\* 校验位长度为 2 个字节，采用 CCITT 标准的 CRC16，其生成多项式为 x16+x12+x5+1，简记为1021

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public class CRC16Util {

public static int getCRC(byte[] bytes){

int crc = 0x0000; // initial value

int polynomial = 0x1021; // 0001 0000 0010 0001 (0, 5, 12)

for (byte b : bytes) {

for (int i = 0; i < 8; i++) {

boolean bit = ((b >> (7-i) & 1) == 1);

boolean c15 = ((crc >> 15 & 1) == 1);

crc <<= 1;

if (c15 ^ bit) crc ^= polynomial;

}

}

crc &= 0xffff;

return crc;

}

}