车牌识别一体机

TCP 通讯协议说明手册

非常感谢您使用我们公司的产品，我们将竭诚为您提供最好的服务。
本手册可能包含技术上不准确的地方或文字错误，欢迎您的纠正。
本手册内容将做定期的更新，更新内容将在本手册的新版本中加入。
我们随时会改进或更新本手册中描述的产品或程序。

版本变更记录

版本号	拟制日期	版本描述	存档编号
1.0.10	2018.10.23	1. 增加NTP配置获取； 2. 增加NTP配置设置；	
1.0.9	2017.03.29	1. 增加获取/设置用户自定义数据 2. 增加获取/设置串口参数 3. 增加获取IO输出状态 4. 获取设备存储信息	
1.0.8	2017.12.12	1. 增加算法识别参数的获取/设置 2. 增加算法识别属性的获取 3. 增加LED控制参数的设置/获取 4. 增加LED控制属性的获取 5. 增加恢复设备默认配置	
1.0.7	2017.11.24	1. 增加设置/获取虚拟线圈参数 2. 增加设置/获取识别区域参数	
1.0.6	2017.10.16	1 设备维护管理相关 2 车牌识别中获取视频播放uri 3 注册脱机事件 4 设备组网相关 5 用户登录相关	
1.0.5	2017.6.16	1 清除组网数据 2 得到当前组网内部所有设备配置 3 获取组网图片	
1.0.4	2016.6.13	1 增加设置、获取网络参数 2 增加设置中心服务器配置	
1.0.3	2016.3.25	Tcp 协议纠错	
1.0.2	2016.2.16	1 新增Linux环境下java的Demo 2 从此版本开始，版本号以三位整数进行发布	
1.0.0.1	2016.1.12	1 开发手册对接口说明，进行分类，方便用户查看	

目录

1 基本说明

设备作为服务端，通过 TCP 协议将识别结果发送到上位机；上位机作为客户端，使用普通 socket 与设备通信。服务端监听端口默认为 8131。

客户端先使用 socket 主动连接 tcp server 服务端（端口号 8131），连接成功后，便可使用如下一系列的 tcp 协议操作我们的设备，以满足用户的需求。

1.1 收发命令包与数据包格式

0	1	2	3	4	5	6	7	标尺
'V'	'Z'	包类型	包序号	包数据长度				数据头
Data								数据

上表是一个标准的 TCP 数据包的格式，数据主要分为两部分：一个 8 个字节的数据包头。接下来是数据，数据长度在前面的包头里面。

接下来我们重点介绍数据包头，数据包头一共 8 个字节，其含义如下：

- **字节位 0**、 ASCII 字符'V'，整数 86。
- **字节位 1**、 ASCII 字符'Z'，整数 90.
- **字节位 2**、 Data 数据类型
  - **0X00** 数据包。
  - **0X01** 心跳包，数据长度为 0， DATA 为空
- **字节位 3**、 包序号，包序列号为递增的数值，用于对应请求命令与返回命令；例如发送一个请求命令到服务器端，服务器端在返回结果时，会将请求命令中的包序列号填充到返回数据包的包序列号中，便于客户端这边将返回结果与请求命令进行对应。如果不存在对应问题，则设置为 0 即可。如果编号到 255，则从 0 开始重新编号。
- **字节位 4－7**、一共四位，代表接下来的数据长度。这个数据是网络字节序，在接收的时候务必调用 htol 这样的函数将网络字节序转换成主机字节序。数据的长度不要大于 1024 * 1024 = 1MB。服务器将不会接收超过 1MB 大小的数据，同样服务器也不会发送超过 1MB 大小的数据包。理论上，除了心跳包之外，其它所有数据的长度都不会是 0。
- **DATA 数据位**、根据前面的包头，有不同的数据。

1.2 具体命令

命令采用 json 格式，注意不论请求还是回复，所有的 JSON 字符全部使用小写字符。协议通用格式如下：

```
request:
{
    "cmd":"xxx"
    "body":"{}"
}
```

```
response:
{
    "cmd":"xxx",
    "id":"1234",
    "state_code":200
    "body":"{}"
}
```

字段名	说明

cmd	命令字符串
body	传递的数据，可为空
state_code	返回结果状态，状态定义如下

状态码定义：(参照 http 的返回状态码定义的)
200　请求的命令执行成功
400　客户端请求出错（因为错误的语法导致服务器无法理解请求信息）
401　客户没有权限，请求需要用户验证
404　请求的资源不存在
405　请求的命令不存在
408　请求超时
500　服务器内部错误

2 Tcp 协议命令

2.1 设备维护管理

2.1.1 获取设备的序列号: getsn

请求命令格式

```
{
    "cmd": "getsn",
    "id":"123456"
}
```

结果返回值

如果一切正常则返回如下数据：

```
{
    "cmd": "getsn",
    "id":"123456",
    "state_code":200,
   "value": "XXXXXXXX-XXXXXXXX"
}
```

如果设备内部有问题，则会返回如下数据：

```
{
   "cmd": "getsn",
   "value": "unknown"
}
```

字段名	说明
value	设备序列号：正确值为 17 位长的字符串，前 8 位 + '-' + 后 8 位

2.1.2 获取设备的硬件版本信息: get_hw_board_version

请求命令格式

```
{
    "cmd":"get_hw_board_version",
    "id" : "12"
}
```

结果返回值

如果一切正常则返回如下数据：

```
{
```

```
    "body":{
        "board_version":3435973836,
        "exdataSize":3435973836
     },
    "cmd":"get_hw_board_version",
    "id":"12",
    "state_code":200
}
```

如果设备内部有问题，返回如下数据：

```
    {
                "body" : {
                    "board_version" : ,
                    "exdataSize" :
                 },
                 "cmd" : "get_hw_board_version",
                 "id" : "12",
                 "state_code" : 500
    }
```

字段名	说明
board_version	硬件版本号
exdataSize	TODO
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
state_code:	状态码 ● 200：成功 ● 400：客户端请求出错 ● 500：服务器内部错误（内部设置错误）

2.1.3 获取设备当前时间戳: get_device_timestamp

请求命令参数

```
{
    "cmd": "get_device_timestamp",
    "id" : "999999"
}
```

结果返回

```
{
    "cmd": "get_device_timestamp",
    "id" : "999999",
    "state_code" : 200,
    "timestamp": 1443213456
}
```

字段名	说明
timestamp	时间（格林威治时间，单位秒）

2.1.4 设置系统时间：set_time

请求命令参数

```
{
    "cmd" : "set_time",
    "id" : "132156",
   "timestring" : "2015-03-17 20:47:02"
}
```

结果返回

```
{
    "cmd" : "set_time",
```

```
    "id" : "132156",
    "state_code" : 200
}
```

字段名	说明
state_code:	状态码 ● 200：成功 ● 400：客户端请求出错（timestring 错误） ● 500：服务器内部错误（内部设置错误）
timestring	● 时间字符串，格式必须是： “XXXX-XX-XX XX:XX:XX”

2.1.5

2.1.6 设置网络参数：set_networkparam

请求命令参数

```
{
    "cmd" : "set_networkparam",
    "id" : "132156",
    "body":{
            "ip":"192.168.1.177",
            "netmask":"255.255.255.0",
            "gateway":"192.168.1.1",
            "dns":"0.0.0.0"
    }
}
```

结果返回

```
{
    "cmd" : "set_networkparam",
    "id" : "132156",
    "error_msg" : "success",
    "state_code" : 200
}
```

字段名	说明
state_code:	状态码 ● 200：成功 ● 500：服务器内部错误
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
body	需要传递的网络数据（ip ,netmask ,gateway ,dns 字段均为可选的，即需要设置哪一项数据就列出哪一项）
ip	Ip 地址
netmask	子网掩码
gateway	网关
dns	dns 服务器

2.1.7 获取网络参数：get_networkparam

请求命令参数

```
{
    "cmd" : "get_networkparam",
    "id" : "132156"
}
```

结果返回

```
{
    "body" : {
        "dhcp_enable" : 0,
        "dns" : "0.0.0.0",
        "gateway" : "192.168.3.1",
        "ip" : "192.168.3.48",
        "netmask" : "255.255.255.0"
    },
    "cmd" : "get_networkparam",
    "error_msg" : "success",
    "id" : "132156",
    "state_code" : 200
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
body	获取到的网络数据
ip	Ip 地址
netmask	子网掩码
gateway	网关
dns	Dns 服务器

2.1.8 设置中心服务器网络参数：set_centerserver_net

请求命令参数

```
{
    "cmd" : "set_centerserver_net",
    "id" : "132156",
    "body":{
        "hostname":"192.168.1.106",
        "port":80,
        "enable_ssl":false,
        "ssl_port":443,
        "http_timeout":5
    }
}
```

结果返回

```
{
    "cmd" : "set_centerserver_net",
    "id" : "132156",
    "state_code" : 200
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30
body	需要传递中心服务器参数数据
hostname	中心服务器地址
port	中心服务器端口
enable_ssl	开启 ssl 连接
ssl_port	ssl 端口
http_timeout	超时时间

2.1.9 设置当前配置为用户默认配置：set_user_default_cfg

请求命令参数

```
{
   "cmd" : "set_user_default_cfg",
   "id" : "132156"
}
```

结果返回

```
{
   "cmd" : "set_user_default_cfg",
    "id" : "132156",
    "error_msg" : "success",
    "state_code" : 200
}
```

字段名	说明
state_code:	状态码 ● 200：成功 ● 400：客户端请求出错 ● 500：服务器内部错误（内部设置错误）
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
error_msg	错误信息，失败时返回错误原因；

2.1.10 修改设备 admin 密码：set_adminpass

请求命令参数

```
{
     "cmd" : "set_adminpass",
      "id" : "132156",
      "body":{
           "old_pass":"asgwe4AGSAD45",
           "new_pass":"fdas213asfdgad"
       }
}
```

结果返回

```
{
   "cmd" : "set_adminpass",
    "id" : "132156",
    "state_code" : 200,
    "error_msg" :"success"
}
```

字段名	说明
state_code:	状态码 ● 200：成功 ● 400：客户端请求出错 ● 500：服务器内部错误（内部设置错误）
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
old_pass	经过 64 位编码后的 Admin 旧密码；
new_pass	经过 64 位编码后的 Admin 新密码；

2.1.11 重启设备:reboot_dev

请求命令参数

```
{
	"cmd" : "reboot_dev",
	"id" : "132156"
}
```

结果返回

```
{
	"cmd" : "reboot_dev",
	"id" : "132156",
	"state_code" : 200,
}
```

且设备重启

字段名	说明
state_code:	状态码 ● 200：成功 ● 400：客户端请求出错 ● 500：服务器内部错误（内部设置错误）
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id

2.1.12 建安顺 APP 过期设置：jasauthtime，jasauthtime_rsp

请求命令参数

```
{
	"cmd" : "jasauthtime",
	"id" : "132156",
	"year": 2017,
	"mon":12,
	"day":12,
	"hour":13,
	"min":14,
	"sec":49
}
```

结果返回

当发送 CMD 为：jasauthtime_rsp 的消息时有如下回复：

```
{
	"cmd" : " jasauthtime_rsp ",
	"id" : "132156",
	"state_code" : 200,
}
```

字段名	说明
state_code:	状态码 ● 200：成功 ● 400：客户端请求出错 ● 500：服务器内部错误（内部设置错误）
year	年
mon	月
day	日
hour	时

min	分钟
sec	秒

2.1.13 对 DM8127 的单板：获取降噪模式和降噪强度：get_denoise

请求命令参数

```
{
     "cmd" : "get_denoise",
      "id" : "132156"
}
```

结果回复

```
{
   "cmd" : "get_denoise",
    "id" : "132156",
    "state_code" : 200,
    "error_msg" :"success",
    "body":
          {
             "mode": 1,
             "strength": 10
     }
 }
```

字段名	说明
mode	降噪模式
strength	降噪强度
state_code:	状态码 ● 200：成功 ● 400：客户端请求出错 ● 500：服务器内部错误（内部设置错误）
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id

2.1.14 对 DM8127 的单板：设置降噪模式和降噪强度：set_denoise

请求命令参数

模式设置：

```
{
    "cmd" : "set_denoise",
    "id" : "132156",
    "body":{
         "mode": 1
     }
}
```

强度设置：

```
{
     "cmd" : "set_denoise",
      "id" : "132156",
      "body":
      {
           "strength": 0
       }
}
```

结果返回

```
{
     "cmd" : "set_denoise",
     "id" : "132156",
```

```
        "state_code" : 200,
        "error_msg" :"success"
    }
```

服务器内部错误

```
    {
        "cmd" : "set_denoise",
          "id" : "132156",
          "state_code" : 500,
          "error_msg" :"service inner error"
    }
```

字段名	说明
mode	降噪模式
strength	降噪强度
state_code:	状态码 ● 200：成功 ● 400：客户端请求出错 ● 500：服务器内部错误（内部设置错误）
id	序列字符串（唯一），字符串长度 ＜ 30，等于请求时的 id

2.1.15 开始自动聚焦：startfocusandzoom

请求命令参数

```
{
"cmd" : "startfocusandzoom",
"id" : "1548",
   "body":{
          "value" : 1
     }
}
```

结果返回

```
{
    "cmd" : "startfocusandzoom",
    "id" : "1548",
    "state_code" : 200
}
```

字段名	说明
value	自动变倍/调焦 ● 0 停止 ● 1 focus 焦增加 ● 2 focus 焦减小 ● 3 zoom 变倍增加 ● 4 zoom 变倍减小

2.1.16 停止自动聚焦：stopfocusandzoom

请求命令参数

```
{
    "cmd" : "stopfocusandzoom",
    "id" : "1548"
}
```

或者

```
{
"cmd" : "startfocusandzoom",
"id" : "1548",
  "body":{
        "value" : 0
    }
}
```

结果返回

```
{
    "cmd" : "stopfocusandzoom",
    "id" : "1548",
    "state_code" : 200
}
```

2.1.17 恢复设备默认配置：set_factorydefault

请求命令参数

```
{
    "cmd" : "set_factorydefault",
    "id" : "1548" ,
  "body":{
        "factorydefault" : 0
    }
}
```

结果返回

```
{
    "cmd" : "set_factorydefault",
    "id" : "1548",
    "state_code" : 200
}
```

字段名	说明
factorydefault	恢复设备默认值 ● 0 完全恢复 ● 1 部分恢复

2.2 车牌识别

2.2.1 配置推送数据方式: ivsresult

请求命令参数

```
{
    "cmd": "ivsresult",
    "id" : "123",
    "enable": true,
    "format": "json",
    "image": true,
    "image_type": 0
}
```

字段名	说明
enable	是否允许推送识别结果，默认值：false ● true 表示允许推送识别结果 ● false 表示不推送识别结果
format	推送识别结果数据格式，默认值：json

	● bin 表示识别结果数据格式为二进制数据格式 ● json 表示识别结果数据格式为 json 数据格式
image	识别结果是否包含图片，默认值：true ● true 表示识别结果包含图片 ● false 表示识别结果不包含图片
image_type	识别的图片类型，默认值：0 ● 0 表示返回识别结果全图 ● 1 表示只返回车牌区域小图 ● 2 表示返回两种图片

当配置完成后（配置为允许推送识别结果的情况 enable：ture），并且在服务器端有 ivsresult 识别结果产生后，tcp server 会主动推送 ivsresult 识别结果给客户端，推送的识别结果格式如下

结果返回值

```
{
    "cmd" : " ivsresult",
    "id" : "123",
    "state_code" : 200
}
```

配置完成后，接收到识别结果为配置值：

正常情况下，一个完整的识别结果（返回 2 种图片的情况）推送消息格式如下：

‘V’	‘Z’	0	0	长度（数据包的大小）
识别结果字符串+字符串结束符‘\0’				
全图的 jpg 格式数据				
小图片（车牌区域图片）的 jpg 格式数据				

例如下面就是一个正常的 JSON 识别推送结果:

```
{
    "PlateResult" : {
        "bright" : 0,
        "carBright" : 0,
        "carColor" : 0,
        "colorType" : 1,
        "colorValue" : 0,
        "confidence" : 99,
        "direction" : 0,
        "license" : "青 PTW3Z3",
        “enable_encrypt”:0,
        "location" : {
            "RECT" : {
                "bottom" : 392,
                "left" : 690,
                "right" : 834,
                "top" : 350
            }
        },
        "timeStamp" : {
            "Timeval" : {
                "sec" : 1458882234,
```

```
                    "usec" : 921325
                }
            },
            "timeUsed" : 0,
            "triggerType" : 1,
            "type" : 1,
            "plate_true_width" : 1,
            "plate_distance" : 50,
            "fake_plate" : 1,
            "car_location" : {
                "RECT" : {
                    "bottom" : 392,
                    "left" : 690,
                    "right" : 834,
                    "top" : 350
                }
            },
            "car_brand" : {
                "brand" : 1,
                "type" : 2,
                "year" : 2017
            },
            "featureCode" : "asdgfb4AGD",
        },
        "active_id" : 0,
        "clipImgSize" : 1103,
        "cmd" : "ivs_result",
        "fullImgSize" : 51566,
        "id" : 0,
        "imageformat" : "jpg",
        "timeString" : "2016-03-25 13:03:54"
}
```


其中各字段的含义如下表所示：

字段名	说明
license	车牌号码（汉字为 GB2312 编码）
enable_encrypt	车牌是否加密（0 不加密，1 加密）
colorValue	车牌颜色
colorType	车牌颜色序号，详见车牌颜色定义 LC_X
type	车牌类型，详见车牌类型定义 LT_X
confidence	车牌可信度
bright	亮度评价
direction	运动方向，详见运动方向定义 DIRECTION_X
location：rect	车牌位置
timeUsed	识别所用时间
carBright	车的亮度
carColor	车的颜色，详见车辆颜色定义 LCOLOUR_X
timeStamp	识别时间点
triggerType	触发结果的类型,见 TH_TRIGGER_TYPE_BIT
cmd	当前指令名称
id	识别记录的编号
imageformat	图片格式
timeString	触发时间字符串，格式如：2015-01-02 03:04:05

fullImgSize	整幅大图的尺寸（字节数）
clipImgSize	车牌区域图片的尺寸（字节数）
active_id	车牌加密方式
plate_true_width	车牌真实宽度
plate_distance	车牌距离
fake_plate	是否是伪车牌
car_location	车辆位置
Car_Brand	车辆品牌信息
featureCode	特征码(16 位字符串)

Car_Brand 中个字段信息

brand	车辆品牌信息
type	车辆类型
year	年份

另，LC_X，LT_X，DIRECTION_X，LCOLOUR_X，TH_TRIGGER_TYPE_BIT 详细定义请参考头文件 VzClientSDK_LPDefine.h。

如果用户设置的是二进制识别结果，那么将会收到如下的二进制数据：

'V'	**'Z'**	**2**	**0**	**长度（整个包的大小）**
'I'	**'R'**	**1**	**0**	**长度（识别结果结构体的大小）**
识别结果结构体(TH_PlateResult)				
'I'	**'R'**	**2（大图片）**	**0**	**长度（大图像数据的大小）**
全图的图片数据(完整的 jpg 格式)				
'I'	**'R'**	**3（小图片）**	**0**	**长度（小图像数据的大小）**
小图片（车牌区域图片）数据(完整的 jpg 格式)				

推送的识别结果结构体定义为 TH_PlateResult，这一个结构体的定义请参考头文件 VzClientSDK_LPDefine.h。

2.2.2 获取最近一次识别结果: getivsresult

请求命令参数

```
{
    "cmd" : "getivsresult",
    "image" : true,
    "format" : "json"
}
```

字段名	说明
image	是否接收识别结果图片，默认值：false ● true 需要识别结果图片 ● false 不需要识别结果图片
format	推送识别结果数据格式，默认值：json ● bin 表示识别结果数据格式为二进制数据格式

	● json 表示识别结果数据格式为 json 数据格式。

结果返回值

这个命令的结果返回值，参见配置推送数据方式里面的识别结果返回值。

2.2.3 手动触发车牌识别：trigger

请求命令参数

```
{
    "cmd": "trigger"

}
```

结果返回值

这个命令的结果返回值，参见配置推送数据方式里面的识别结果返回值。

2.2.4 获取记录最大 id: get_max_rec_id

主要用于在脱机之后，断线之后使用。

请求命令参数

```
{
    "cmd": "get_max_rec_id",
    "id": "123"
}
```

结果返回

```
{
    "cmd": "get_max_rec_id",
    "id": “123”,
    "state_code":200,
    "max_id": 1024
}
```

字段名	说明
max_id	识别结果记录中最大的 id 值

2.2.5 获取历史记录: get_record

根据 id 获取识别记录

请求命令参数

```
{
    "cmd": "get_record",
    "id": 2,
    "format": "json",
    "image": true
}
```

字段名	说明
id	识别结果记录的 id 值
format	推送识别结果数据格式，默认值：json bin 表示识别结果数据格式为二进制数据格式，当前不支持 bin 形式。 json 表示识别结果数据格式为 json 数据格式。
image	识别结果是否包含图片，默认值：true true 表示识别结果包含图片

	false 表示识别结果不包含图片

结果返回

这个命令的结果返回值，参见配置推送数据方式里面的识别结果返回值。

2.2.6 获取记录图片: get_image、get_offline_image

按记录 ID 获取识别结果对应的图片

请求命令参数

```
{
    "cmd": "get_image",
    "id": 2
}
```

字段名	说明
id	识别结果记录的 id，可在识别记录结构体中找到。

结果返回

```
{
    "cmd": "get_image",
    "id": 2,
    "size": 57344,
    "state_code" : 200
}
```

字段名	说明
id	识别结果记录的 id，可在识别记录结构体中找到。前面的获取脱机记录中的 ID 也一样
size	图片的大小

注释：这个 JSON 后面跟了一个 57344 大小的二进制图片，其中 JSON 字符串以’\0’结束。

2.2.7 抓取当前图片 get_snapshot

请求命令参数

```
{
    "cmd":"get_snapshot",
    "id":"123456"
}
```

结果返回

```
{
    "cmd":"get_snapshot",
    "state_code":200,
    "id":"123456",
    "imgformat":"jpg",
    "imgdata":"14ewrfswtergfhbfr=="
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
state_code	状态码
imgformat	图片格式（jpg）
imgdata	图片数据：经过 base64 转码后的数据

2.2.8 获取视频播放的 uri: get_rtsp_uri

请求命令参数

```
{
 "cmd" : "get_rtsp_uri",
  "id" : "132156"
}
```

结果返回

```
{
   "cmd" : " get_rtsp_uri ",
    "id" : "132156",
    "state_code" : 200,
    "uri":"rtsp://192.168.7.7:8557/h264"
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
state_code	状态码
uri	当前播放视频的 URI

2.2.9 获取虚拟线圈参数：get_virloop_para

请求命令参数

```
{
 "cmd" : "get_virloop_para",
  "id" : "132156"
}
```

结果返回

```
{
    "body" : {
        "virtualloop" : {
            "max_plate_width" : 400,
            "min_plate_width" : 45,
            "dir" : 0,
            "trigger_gap" : 10,
            "virtualloop_num" : 1,
            "loop" : [ {
            "id" : 0,
            "enable" : true,
            "point_num" : 4,
             "point" :
                  [ { "x" : 2400, "y" : 12000 },
                    { "x" : 13984, "y" : 12000 },
                    { "x" : 15984, "y" : 14000 },
                    { "x" : 400, "y" : 14000 } ]
                }]
          }
     },
     "cmd" : "get_virloop_para",
     "error_msg" : "success",
     "id" : "132156",
     "state_code" : 200
}
```

Body 中个字段含义

字段名	说明
virtualloop	虚拟线圈
max_plate_width	最大车牌尺寸

min_plate_width	最小车牌尺寸
dir	运动方向
trigger_gap	相同车牌的触发时间间隔
virtualloop_num	虚拟线圈数量

Loop 中字段含义：

字段名	说明
id	线圈 ID
enable	是否可用
point_num	顶点个数
Poin	顶点坐标 ● X 横向坐标，从左至右，从 0 递增 ● Y 纵向坐标，从上至下，从 0 递增

2.2.10 设置虚拟线圈参数：set_virloop_para

请求命令参数

```
{
    "body" : {
        "virtualloop" : {
            "max_plate_width" : 400,
            "min_plate_width" : 45,
            "dir" : 0,
            "trigger_gap" : 10,
            "virtualloop_num" : 1,
            "loop" : [ {
            "id" : 0,
            "enable" : true,
            "point_num" : 4,
             "point" :
                  [ { "x" : 2400, "y" : 12000 },
                    { "x" : 13984, "y" : 12000 },
                    { "x" : 15984, "y" : 14000 },
                    { "x" : 400, "y" : 14000 } ]
                }]
          }
     },
    "cmd" : "set_virloop_para",
    "id" : "132156"
}
```

结果返回

```
{
    "cmd" : "set_virloop_para",
    "id" : "132156",
    "error_msg" : "success",
    "state_code" : 200
}
```

参数含义请参考 2.2.9 上方；

2.2.11 获取虚拟线圈属性：get_virloop_prop

请求命令参数

```
{
 "cmd" : "get_virloop_prop",
  "id" : "132156"
```

```
}
```

结果返回

```
{
    "body" :{
          "virtualloop" :{
               "plate_width" :{
                     "max" : 600,
                      "min" : 45,
                      "default_max" : 400,
                      "default_min" : 25
                },
               "trigger_gap" :{
                      "max" : 255,
                      "min" : 0,
                      "default" : 10
                },
               "dir" :{
                      "default" : 0,
                      "types" :
                           [{"content" : "双向",   "type" : 0},
                           {"content" : "从上至下",   "type" : 1},
                           {"content" : "从下至上",   "type" : 2}]
                }
           }
    }
    "cmd" : "get_virloop_prop",
    "error_msg" : "success",
    "id" : "132156",
    "state_code" : 200
}
```

字段名	说明
plate_width	车牌宽度 ● max 最大宽度 ● min 最小宽度 ● default_max 默认最大宽度 ● default_min 默认最小宽度
trigger_gap	相同车牌的触发时间间隔 ● max 最长间隔时间 ● min 最短间隔时间 ● default 默认间隔时间
dir	运动方向 ● content”双向”, type,0 ● content”从上至下”,type,1 ● content”从下至上”,type2 ● default 默认值

2.2.12 获取识别区域参数：get_reco_para

请求命令参数

```
{
 "cmd" : "get_reco_para",
  "id" : "132156"
}
```

结果返回

```
{
    "body" : {
        "recognition_area" : {
            "polygon_num" : 1,
            "polygon" :
             [ {
                "id" : 1,
                "enable" : true,
                "point_num" : 4,
                "point" :
                    [ { "x" : 4096, "y" : 4096 },
                      { "x" : 12288, "y" : 4096},
                      { "x" : 13926, "y" : 13926 },
                      { "x" : 2457,   "y" : 13926 } ]
             }]
        }
    },
    "cmd" : "get_reco_para",
    "id" : "132156",
    "error_msg" : "success",
    "state_code" : 200
}
```

recognition_area 识别区域中字段含义

字段名	说明
polygon_num	规则数量
enable	是否可用
point_num	顶点个数
Poin	顶点坐标 ● X 横向坐标，从左至右，从 0 递增 ● Y 纵向坐标，从上至下，从 0 递增

2.2.13 设置识别区域参数：set_reco_para

请求命令参数

```
{
    "body" : {
        "recognition_area" : {
            "polygon_num" : 1,
            "polygon" :
             [ {
                "id" : 1,
                "enable" : true,
                "point_num" : 4,
                "point" :
                    [ { "x" : 4096, "y" : 4096 },
                      { "x" : 12288, "y" : 4096},
                      { "x" : 13926, "y" : 13926 },
                      { "x" : 2457,   "y" : 13926 } ]
             }]
        }
    },
    "cmd" : "set_reco_para",
    "id" : "132156"
}
```

结果返回

```
{
```

```
    "cmd" : "set_reco_para",
    "id" : "132156",
    "error_msg" : "success",
    "state_code" : 200
}
```

参数字段含义参见 2.2.13 上方

~~2.2.14 获取识别区域属性：get_reco_prop~~

请求命令参数

```
{
    "cmd" : "get_reco_prop",
    "id" : "132156"
}
```

结果返回

```
{
    "body" : {
        "polygon_num" : { "max" : 1, "min" : 0 },
        "polygon" : [ {
            "id" : 1,
            "enable" : true,
            "point_num" : 4,
            "point" :
                [ { "x" : 4096, "y" : 4096 },
                  { "x" : 12288, "y" : 4096 },
                  { "x" : 13926, "y" : 13926},
                  { "x" : 2457, "y" : 13926 } ]
        } ]
    },
    "cmd" : "get_reco_prop",
    "id" : "132156",
    "error_msg" : "success",
    "state_code" : 200
}
```

字段名	说明
polygon_num	规则数量 ● Max 最大数量 ● Min 最小数量

Polygon 中各字段含义

id	规则 ID
enable	是否可用
point_num	顶点个数
Poin	顶点坐标 ● X 横向坐标，从左至右，从 0 递增 ● Y 纵向坐标，从上至下，从 0 递增

2.2.15 获取算法识别参数： get_alg_result_para

请求命令参数

```
{
    "cmd" : "get_alg_result_para",
    "id" : "132156"
}
```

回复

```
{
    "cmd" : " get_alg_result_para",
    "id" : "132156",
    "state_code" : 200,
    "body" : {
        “snap_resolution” : 0,
        "snap_image_quality" : 100,
        "out_result_type" : 1,
        "recognition_type" : 1,
        "province" : 1,
        "run_mode":0 ,
        "alg_version" : "2016LPRALG",
        "time_zone" : 0,
        "reco_dis" : 0
    }
}
```

字段含义

字段名	说明
snap_resolution	抓拍图片分辨率
snap_image_quality	抓拍图片质量，单位百分比;
out_result_type	输出类型;
recognition_type	识别类型
province	预设省份
run_mode	运行模式
alg_version	算法版本
time_zone	时区
reco_dis	识别距离 0: 1-6 米 1: 大于 6 米

2.2.16 设置算法识别参数： set_alg_result_para

请求命令参数

```
{
    "cmd" : "set_alg_result_para",
    "id" : "132156",
    "body" : {
        "snap_resolution" : 0,
        "snap_image_quality" : 100,
        "out_result_type" : 1,
        "recognition_type" : 1,
        "province" : 1,
        "run_mode":0 ,
        "alg_version" : "2016LPRALG",
        "time_zone" : 0,
        "reco_dis" : 0
    }
}
```

回复

```
{
    "cmd" : "set_alg_result_para",
    "id" : "132156",
    "state_code" : 200
}
```

字段含义

字段名	说明
snap_resolution	抓拍图片分辨率
snap_image_quality	抓拍图片质量，单位百分比;
out_result_type	输出类型;
recognition_type	识别类型
province	预设省份
run_mode	运行模式
alg_version	算法版本
time_zone	时区
reco_dis	识别距离 0: 1-6 米 1: 大于 6 米

2.2.17 获取算法识别参数属性： get_alg_result_prop

请求命令参数

```
{
    "cmd" : "get_alg_result_prop",
    "id" : "132156"
}
```

回复

```
{
    "cmd" : "get_alg_result_prop",
    "id" : "132156",
    "state_code" : 200,
    "body" :
    {
        "snap_resolution" :
        {
            "default" : 10,
            "types" : [
                { "content" : "720*576", "type" : 6 },
                { "content" : "1280*720", "type" : 9 },
                { "content" : "1920*1080", "type" : 10 } ]
        },
        "snap_image_quality" :
        {
            "max" : 100,
            "min" : 10,
            "default" : 70
        },
        "out_result_type" :
        {
            "default" : 16,
            "types" : [
                { "content" : "稳定识别触发", "type" : 1 },
                { "content" : "虚拟线圈", "type" : 2 },
                { "content" : "IO1 触发", "type" : 16},
                { "content" : "IO2 触发", "type" : 32 },
                { "content" : "IO3 触发", "type" : 64 }
            ]
        },
        "recognition_type" :
        {
```

```
            "default" : 60282,
            "types" : [
                    { "content" : "蓝牌", "type" : 1 },
                    { "content" : "黄牌", "type" : 24},
                    { "content" : "黑牌", "type" : 4 },
                    { "content" : "教练车", "type" : 8192},
                    { "content" : "警车", "type" : 32 },
                    { "content" : "武警车", "type" : 32832 },
                    { "content" : "军车", "type" : 768 },
                    { "content" : "港澳", "type" : 18432},
                    { "content" : "使馆车", "type" : 1024}
                ]
        },
        "province" :
        {
            "default" : 0,
            "types" : [
              { "content" : "无", "type" : 255 },
              { "content" : "京", "type" : 0 },
              { "content" : "津", "type" : 1 },
              { "content" : "冀", "type" : 2 },
              { "content" : "晋", "type" : 3 },
              { "content" : "蒙", "type" : 4 },
              { "content" : "辽", "type" : 5 },
              { "content" : "吉", "type" : 6 },
              { "content" : "黑", "type" : 7 },
              { "content" : "沪", "type" : 8 },
              { "content" : "苏", "type" : 9 },
              { "content" : "浙", "type" : 10 },
              { "content" : "皖", "type" : 11 },
              { "content" : "闽", "type" : 12 },
              { "content" : "赣", "type" : 13 },
              { "content" : "鲁", "type" : 14 },
              { "content" : "豫", "type" : 15 },
              { "content" : "鄂", "type" : 16 },
              { "content" : "湘", "type" : 17 },
              { "content" : "粤", "type" : 18 },
              { "content" : "桂", "type" : 19 },
              { "content" : "琼", "type" : 20 },
              { "content" : "渝", "type" : 21 },
              { "content" : "川", "type" : 22 },
              { "content" : "贵", "type" : 23 },
              { "content" : "云", "type" : 24 },
              { "content" : "藏", "type" : 25 },
              { "content" : "陕", "type" : 26 },
              { "content" : "甘", "type" : 27 },
              { "content" : "青", "type" : 28 },
              { "content" : "宁", "type" : 29 },
              { "content" : "新", "type" : 30 },
              { "content" : "港", "type" : 31 },
              { "content" : "澳", "type" : 32 },
              { "content" : "台", "type" : 33 },
              { "content" : "警", "type" : 34 },
              { "content" : "使", "type" : 35 },
              { "content" : "WJ", "type" : 36 },
              { "content" : "领", "type" : 37 },
              { "content" : "学", "type" : 38 } ]
          },
        "run_mode" :
            {
```

```
                "default" : 0,
                "types" : [
                        { "content" : "默认", "type" : 0 },
                        { "content" : "停车场", "type" : 1 },
                        { "content" : "卡口", "type" : 2 } ]
            },
        "time_zone" : { "default" : 0 },
        "reco_dis" :
            {
              "default" : 0,
               "types" : [
                        { "content" : "2-4m", "type" : 0 },
                        { "content" : "6-8m", "type" : 1 },
                        { "content" : "4-6m", "type" : 2}
               ]
            }
      }
}
```

2.2.18 设置 LED 参数： set_led_para

请求命令参数

```
{
    "body" : {
        "led_level" : 2,
        "led_mode" : 3,
        "time_ctrl" : [
            {
                "id" : 0,
                "led_level" : 2,
                "time_begin" : "00:00:00",
                "time_end" : "02:03:34",
                "timectrl_enable" : true
            },
            {
                "id" : 1,
                "led_level" : 6,
                "time_begin" : "02:03:34",
                "time_end" : "09:02:23",
                "timectrl_enable" : true
            }
        ]
    },
    "cmd" : "set_led_para",
    "id" : "132156"
}
```

回复

```
{
    "cmd" : "set_led_para",
    "id" : "132156",
    "state_code" : 200
}
```

字段名	说明
led_mode	LED 模式 ● 0 智能 ● 1 常亮

	● 2 常灭 ● 3 时间段
led_level	LED 等级[0,5]
time_ctrl	时间段控制
id	时间段 ID
time_begin	时间段起始时间
time_end	时间段结束时间
timectrl_enable	时间段使能（0 去使能 1 使能）

2.2.19 获取 LED 参数： get_led_para

请求命令参数

```
{
    "cmd" : "get_led_para",
    "id" : "132156"
}
```

回复

```
{
    "body" : {
        "led_level" : 2,
        "led_mode" : 3,
        "time_ctrl" : [
            {
                "id" : 0,
                "led_level" : 2,
                "time_begin" : "00:00:00",
                "time_end" : "01:02:23",
                "timectrl_enable" : true
            },
            {
                "id" : 1,
                "led_level" : 6,
                "time_begin" : "01:02:24",
                "time_end" : "09:02:23",
                "timectrl_enable" : true
            }
        ]
    },
    "cmd" : "get_led_para",
    "id" : "132156",
    "state_code" : 200
}
```

2.2.20 获取 LED 属性： get_led_prop

请求命令参数

```
{
    "cmd" : "get_led_prop",
    "id" : "132156"
}
```

回复

```
{
    "cmd" : " get_led_prop ",
    "id" : "132156",
    "state_code" : 200,
    "body" :
```

```
    {
        "led_mode" :
            {
                "default" : 0,
                 "types" : [
                    { "content" : "智能", "type" : 0 },
                    { "content" : "常亮", "type" : 1 },
                    { "content" : "常灭", "type" : 2 },
                    { "content" : "时间段", "type" : 3 }
                 ]
            },
         "led_level" :
             {
                "default" : 4,
                "max" : 5,
                "min" : 0
              }
     },
}
```

2.3 设备硬件参数配置

2.3.1 控制 IO 输出: ioctl, ioctl_resp

请求命令参数示例

```
{
    "cmd" : "ioctl",
    "id" : "132156",
    "delay" : 500,
    "io" : 0,
    "value" : 2
}
```

```
{
    "cmd" : "ioctl_resp",
    "id" : "132156",
    "delay" : 500,
    "io" : 0,
    "value" : 2
}
```

字段名	说明
io	对应的输出 IO 编号，必需值，正常情况下只有 0，1 两个输出编号
value	输出 IO 的状态值 ● 0 断 ● 1 通 ● 2 先通后断（一般做开闸使用）
delay	先通后断的延迟时间

结果返回

当发送 CMD 为：ioctl_resp 时有如下返回：

```
    {
       "cmd" : " ioctl_resp ",
        "id" : "132156",
        "state_code" : 200
    }
```

2.3.2 获取 IO 输入状态: get_gpio_value

请求命令参数示例

```
{
    "cmd" : "get_gpio_value",
    "id" : "123",
    "gpio" : 0
}
```

结果返回

```
{
    "cmd" : "get_gpio_value",
    "id" : "123",
    "state_code" : 200,
    "gpio" : 0,
    "value" : 1
}
```

字段名	说明
gpio	对应的输入 IO 编号，必需值，正常情况下只有 0，1 两个输入编号
value	输入 IO 的状态值 ● 0 低电平 ● 1 高电平

2.3.3 获取 IO 输出状态: get_gpio_out_value

请求命令参数示例

```
{
    "cmd" : "get_gpio_out_value",
    "id" : "123",
    "gpio" : 0
}
```

结果返回

```
{
    "cmd" : "get_gpio_out_value",
    "id" : "123",
    "state_code" : 200,
    "gpio" : 0,
    "value" : 1
}
```

字段名	说明
gpio	对应的输入 IO 编号，必需值，正常情况下只有 0，1 两个输入编号
value	输入 IO 的状态值 ● 0 低电平 ● 1 高电平

2.3.4 自动聚焦: auto_focus, auto_focus_rsp

请求命令参数

```
{
    "cmd": "auto_focus"
```

```
}
```

```
{
    "cmd": "auto_focus_rsp",
    "id" : "123"
}
```

结果返回

可观察到设备有自动聚焦的动作发生

当 cmd 为：auto_focus_rsp 时，返回如下：

```
{
    "cmd": " auto_focus_rsp ",
    "id" : "123",
    "state_code" : 200
}
```

2.3.5 获取存储设备信息: get_diskinfo

请求命令参数

```
{
"cmd" : "get_diskinfo",
"id":"123"
}
```

结果返回

```
{
    "body" : [
        {
            "devname" : "/dev/mmcblk0",
            "devparts" : [
                {
                    "formatpercent" : 0,
                    "partname" : "/dev/mmcblk0p1",
                    "partspace" : {
                        "left" : 7602,
                        "total" : 7932,
                        "used" : 330
                    },
                    "partstate" : 5
                }
            ],
            "devtotal" : 7948,
            "devtype" : 0
        }
    ],
    "cmd" : "get_diskinfo",
    "error_msg" : "success",
    "id" : "123",
    "state_code" : 200
}
```

字段名	说明
devname	当前磁盘名称
formatpercent	格式化百分比，此值不用关注
left	剩余磁盘空间
total	磁盘总空间
used	已使用磁盘空间

partstate	分区状态 0. 错误 1. 未格式化 2. 未挂载 3. 正在格式化 4. 正在删除文件 5. 正常工作 6. 只读 7. 慢速磁盘 8. 磁盘损坏
devtype	磁盘类型 0. SD 卡 1. HD 卡
Devtotal	设备总内存

2.3.6 获取串口参数: get_serial_para

请求命令参数

```
{
    "cmd" : "get_serial_para",
    "id":"123",
    "serial_port":0
}
```

结果返回

```
{
    "body" : {
        "baud_rate" : 9600,
        "data_bits" : 8,
        "parity" : 0,
        "stop_bits" : 1
    },
    "cmd" : "get_serial_para",
    "id" : "123",
    "serial_port" : 1,
    "state_code" : 200
}
```

字段名	说明
serial_port	串口号
baud_rate	波特率： 2400/4800/9600/19200/38400/57600/115200
data_bits	数据位：固定 8
parity	校验位： 0． 无校验 1． 奇校验 2． 偶校验
stop_bits	停止位： 1 停止位 1

	2　停止位 2

2.3.7　设置串口参数: set_serial_para

请求命令参数

```
{
    "body" : {
        "baud_rate" : 37100,
        "data_bits" : 8,
        "parity" : 1,
        "stop_bits" : 2
    },
    "cmd" : "set_serial_para",
    "id" : "123",
    "serial_port" : 0
}
```

结果返回

```
{
    "cmd" : "set_serial_para",
    "error_msg" : "Sucess",
    "id" : "123",
    "state_code" : 200
}
```

2.4　设备连接

2.4.1　通知在线消息:response_online, response_online_rsp

请求命令参数

```
{
     "cmd": "response_online",
     "id" : "123"
}
```

```
{
     "cmd": "response_online_rsp",
     "id" : "123"
}
```

结果返回

当发送 cmd 为：response_online_rsp 时有如下返回：

```
{
     "cmd": "response_online",
     "id" : "123",
     "state_code" : 200
}
```

2.5　配置透明通道: ttransmission

请求命令参数

```
{
     "cmd": "ttransmission",
     "id" : "999999",
```

```
    "subcmd": "send",
    "datalen": 6,
    "data": "QUJDREVG",
    "comm": "rs485-3"
}
```

```
{
    "cmd": "ttransmission",
    "id" : "999999",
    "subcmd": "init",
    "data": "rs485-3"
}
```

```
{
    "cmd": "ttransmission",
    "id" : "999999",
    "subcmd": "uninit"
}
```

字段名	说明
subcmd	子命令 ● init 初始化 ● uninit 取消初始化 ● send 发送数据
datalen	数据长度，实际数据长度（即编码前的数据长度）
data	字符串数据，经过 base64 编码后的数据
comm	comm 口 ● rs485-1 ● rs485-2

结果返回

如果初始化错误，则返回:

```
{
    "cmd": "ttransmission",
    "id" : "999999",
    "subcmd": "init",
    "state_code" : 400,
    "response": "failed"
}
```

如果初始化成功，则返回:

```
{
    "cmd": "ttransmission",
    "id" : "999999",
    "state_code" : 200,
    "subcmd": "init",
    "response": "ok"
}
```

如果取消初始化失败，则返回

```
{
    "cmd": "ttransmission",
    "id" : "999999",
    "subcmd": "uninit",
    "state_code" : 400,
    "response": "failed"
```

```
}
```

如果取消初始化成功，则返回:

```
{
    "cmd": "ttransmission",
    "id" : "999999",
    "state_code" : 200,
    "subcmd": "uninit",
    "response": "ok"
}
```

发送数据没有任何返回值。

字段名	说明
subcmd	子命令 ● init 初始化 ● uninit 取消初始化 ● send 发送数据
response	返回结果

2.6 白名单

2.6.1 脱机检查

2.6.1.1 注册脱机功能: reg_offline_check

将当前 TCP 客户端注册为脱机检测的响应终端。

请求命令参数示例

```
{
    "cmd" : "reg_offline_check",
    "id" : "999999",
    "interval" : 2
}
```

结果返回

```
{
    "cmd": "offline",
    "id" : "999999",
    "state_code" : 200,
    "response": "ok"
}
```

字段名	说明
interval	脱机响应的超时时间（秒为单位），如果超时未响应，则设备转为脱机状态。
response	返回结果

2.6.1.2 取消脱机注册: reg_offline_check

取消脱机功能

```
    {
    "cmd" : "reg_offline_check",
    "id" : "999999",
    "sucmd" : "cancel"
```

```
    }
```

无结果返回

2.6.1.3 注册脱机事件：register_offline_event

请求命令参数示例

```
{
    "cmd" : "register_offline_event",
    "id" : "132156",
    "body":
     {
         "register_status": 1
     }
}
```

结果返回

```
{
    "cmd" : "register_offline_event",
     "id" : "132156",
     "state_code" : 200,
    "error_msg":"Sucess"
}
```

未注册脱机，返回错误：

```
{
     "cmd" : "register_offline_event",
     "id" : "132156",
     "state_code" : 400,
     "error_msg":"The session do not reg OfflineCheck,please reg it first"
}
```

当脱机状态改变且 register_status!=0 时，收到消息：

```
{
     "cmd" : "offline_status_change",
     "body":
     {
         "offline_status": 0
     }
}
```

字段名	说明
register_status	注册状态，当发生改变且 register_status!=0 时，会收到改变响应消息；

备注：注册脱机事件之前，需要先注册脱机功能，否则失败；

2.6.1.4 获取脱机状态：get_offline_status

请求命令参数示例

```
{
    "cmd" : "get_offline_status",
    "id" : "132156"
}
```

结果回复：

```
{
    "cmd" : "get_offline_status",
```

```
    "id" : "132156",
    "state_code" : 200,
    "body":
     {
         "offline_status": 1
     }
}
```

字段名	说明
offline_status	当前脱机状态；0：脱机，1：在线

2.6.1.5 注册 OpenSDK 监听推送 : register_push_channel

请求命令参数示例

```
{
   "cmd" : "register_push_channel",
    "id" : "132156",
    "body":{
        "register_status": 1
     }
}
```

结果返回

```
{
    "cmd" : "register_push_channel",
    "id" : "132156",
    "state_code" : 200,
   "error_msg" :"Success"
}
```

有推送消息且 register_status!=0 时，会收到推送消息：

```
{
   "cmd" : "opensdk_push_message",
    "body":{
        XXXXXXX
     }
}
```

字段名	说明
register_status	注册 openSDK 状态，当有推送消息且 register_status!=0 时，会收到推送消息；

2.6.1.6 请求 OpenSDK Push : push_msg_to_opensdk

请求命令参数示例

```
{
   "cmd" : "push_msg_to_opensdk",
    "id" : "132156",
    "body":{
        Xxxxxxx
     }
}
```

结果返回

```
{
  "cmd" : "push_msg_to_opensdk",
  "id" : "132156",
```

```
    "state_code" : 200
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
state_code	状态码

2.6.2 操作白名单：white_list_operator

2.6.2.1 增加或者更新白名单：update_or_add

请求命令参数示例

```
{
    "cmd": "white_list_operator",
    "id":"123",
    "operator_type": "update_or_add",
    "dldb_rec": {
        "create_time": "2015-10-10 12:30:40",
        "enable_time": "2015-10-20 12:30:40",
        "overdue_time": "2016-10-20 12:30:40",
        "enable": 1,
        "plate": "京 A12345",
        "time_seg_enable": 1,
        "seg_time": "2016-10-20 12:30:40",
        "need_alarm": 1,
        "vehicle_code": "3254ASFDSFSD",
        "vehicle_comment": "HELOO woradf",
        "customer_id": 144413212
    }
}
```

先根据车牌号进行查找，如果在白名单中找到数据，则为更新记录。如果没有在白名单中找到数据则为添加数据。
其中时间要特别注意，必须是标准的格式：“XXXX-XX-XX XX:XX:XX”。这样才能够和数据库相互对应。

字段名	说明
operator_type	子命令 ● update_or_add 增加或更新白名单 ● delete 删除白名单 ● Select 白名单查询
create_time	白名单创建时间
enable_time	白名单生效时间
overdue_time	白名单失效时间
enable	是否启动这条规则
plate	车牌号
time_seg_enable	是否启用时间段，如果是，那么下面 seg_time，就会生效
seg_time	如果上面启动了时间段。这个字段就开始生效了。最长 1024 个字符
need_alarm	是否需要报警
vehicle_code	用户自定义代码，是一个字符串。最长 32 个字符，不允许重复
vehicle_comment	用户自定义的注释，也是一个字符串。最长 32 个字符
customer_id	用户自己定义 ID，是一个整数，数据库内不保证唯一性

增加或者更新返回值:

```
{
    "cmd": "white_list_operator",
    “id”:”123”,
    state_code" : 200,
    "operator_type": "update_or_add",
    "state": "succeed"
}
```

字段名	说明
operator_type	子命令 ● update_or_add 增加或更新白名单 ● delete 删除白名单 ● select 白名单查询
state	返回结果

2.6.2.2 以车牌号查询白名单：plate

请求命令参数示例

```
{
    "cmd" : "white_list_operator",
    "id" : "999999",
    "operator_type" : "select",
    "plate" : "川 A07273",
    "sub_type" : "plate"
}
```

这个查询是模糊查询，如果车牌号为空的话，那么就会查询到所有的记录。如果只有一部分，那么就可能查询到相似的车牌信息。

字段名	说明
operator_type	子命令 ● update_or_add 增加或更新白名单 ● delete 删除白名单 ● Select 白名单查询
sub_type	子子命令 ● plate 根据车牌号进行查询
plate	车牌号

以车牌号查询白名单返回结果：

```
{
    "cmd" : "white_list_operator",
    "id" : "999999",
    "state_code" : 200,
    "dldb_rec" : [
            {
               "create_time" : "2016-03-25 14:23:58",
               "customer_id" : 144413212,
               "enable" : 6,
               "enable_time" : "2015-12-25 11:00:00",
```

```
                "index" : 1,
                "need_alarm" : 1,
                "overdue_time" : "2015-1-20 11:00:00",
                "plate" : "川 A07273",
                "seg_time" : "2015-1-20 11:00:00",
                "time_seg_enable" : 1,
                "vehicle_code" : "1",
                "vehicle_comment" : "HELOO woradf"
            }
        ],
        "operator_type" : "select",
        "state" : "succeed"
}
```

字段名	说明
operator_type	子命令 ● update_or_add 增加或更新白名单 ● delete 删除白名单 ● select 白名单查询
index	此条记录在数据库中的 id
create_time	白名单创建时间
enable_time	白名单生效时间
overdue_time	白名单失效时间
enable	是否启动这条规则
plate	车牌号
time_seg_enable	是否启用时间段，如果是，那么下面 seg_time，就会生效
seg_time	如果上面启动了时间段。这个字段就开始生效了。最长 1024 个字符
need_alarm	是否需要报警
vehicle_code	用户自定义代码，是一个字符串。最长 32 个字符，不允许重复
vehicle_comment	用户自定义的注释，也是一个字符串。最长 32 个字符
customer_id	用户自己定义 ID，是一个整数，数据库内不保证唯一性
state	返回结果状态

2.6.2.3 白名单删除：delete

请求命令参数示例

```
{
    "cmd":"white_list_operator",
    "id" : "999999",
    "operator_type":"delete",
    "plate ":"京 A12345"
}
```

通过匹配车牌号删除白名单，如果车牌号为空，则清空整个白名单，否则只删除当前匹配成功的记录。

白名单删除返回结果：

```
{
    "cmd":"white_list_operator",
    "id" : "999999",
    "operator_type":"delete",
    "state":"succeed",
```

```
    "state_code" : 200
}
```

字段名	说明
operator_type	子命令 ● update_or_add 增加或更新白名单 ● delete 删除白名单 ● select 白名单查询
state	返回结果状态
plate	车牌号

2.7 识别结果加密

从安全角度考虑或者不希望第三方获取到一些信息，可调用如下接口对识别结果加密。

当前加密功能一共提供了四个接口 get_ems 、 reset_encrypt_key 、 change_encrypt_key、enable_encrypt。用户如果使用这些接口传输自己的明文密码的话，将会非常的不安全，所有的信息都是以明文的形式进行传输。恶意用户很容易通过抓包就能够得到设备的加密密码甚至主密码，所以对这几个接口传输的密码信息进行了加密处理。

接口中部分字段的解释：

active_id：这一个字段指明当前选择的加密字段为多少。其中如果为 0 则为不加密，如果为其它的则为加密

signature：是一个十六位的字符串，全部的值以`0 - 9 A - Z a -z`字符构成。用户在未来的设置密码的时候，应该将这个字符串作为“消息”并结合秘钥（encrpty_key 或者 prime_key）使用`HMAC-SHA1`算法进行加密，再进行`Base64`编码。然后传输给设备，设备才能够开通或者关闭加密或者修改密码或者重置密码。调用加密功能的这四个接口都会得到一个 signature，且每一次用户得到的值都会不一样。signature 一旦更新了之后，用户必须确保使用新的 signature，才能够进行接下来的认证。

注意：用户密码长度应该限制在[4,16]位之前，不然会失败

可借助如下网址中的加密工具来验证设备的加密功能相关接口的正确性

HMAC-SHA 算法加密：http://1024tools.com/hmac

Base64 编码：http://tomeko.net/online_tools/hex_to_base64.php?lang=en

AES 128 位算法：http://aes.online-domain-tools.com/

识别结果加密一般流程：

1. 获取设备支持的加密方式：get_ems
2. 用户可修改密码：对出厂时的默认用户密码进行修改或者对已经修改过的用户密码再次修改：change_encrypt_key
3. 用户可重置密码：用户在忘记用户密码时也可根据主密钥重新设置用户密码：主密钥需要向我们销售要：reset_encrypt_key
4. 根据用户密码开启对识别结果的加密：enable_encrypt

使用主密钥+用户密码加密的原因：如果只使用一个密码（用户密码），当用户忘记这个密码时，需要重置成默认密码，假如默认密码是“000000”，那么代表着所有设备重置一下，密码都变为“000000”。大家都知道这个情况后，设备也将不再安全。例如 A 用户的设备，为了防止 B 用户获取到 A 用户设备上的一些信息，A 用户对他的设备返回结果加了密，B 用户在不知道 A 用户密码的情况下，无法解密加密后的文件，就无法获取到有用的信息。但是当 B 知道了设备重置一下，密码就会变成“000000”，那么 B 用户可向 A 用户的设备发送重置密码命令，然后再修改成自己的密码，就可用自己的密码解密加密后的文件，就获取到了 A 用户设备上的有用信息。主密钥由我们统一管理，且每台设备的主密钥唯一

2.7.1 获取加密方式：get_ems

请求命令参数

```
{
    "cmd" : "get_ems",
    "id" : "999999"
}
```

结果返回

```
{
    "active_id" : 1,
    "cmd" : "get_ems",
    "ems" : [
            {
               "m_id" : 0,
               "name" : "none encrypt"
            },
            {
               "m_id" : 1,
               "name" : "aes-128-ecb-padding-zeros"
            }
     ],
    "signature" : "BNjnUHLkgPaLH3sl",
    "id" : "999999",
    "state_code" : 200
}
```

字段名	说明
active_id	当前加密方式的索引
ems	加密方式列表 ● **m_id** 加密方式的索引 ● **name** 加密方式的名字
signature	一个十六位的字符串（用于对传输的密码进行加密，每调用一次 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt 四个接口中的任意一个接口, signature 都会变，重置/修改密码和开启加密需采用最新的 signature ）
state_code:	状态码 ● 200：成功

2.7.2 获取用户密码：get_encrypt_key

用户根据主密钥获取用户密码；

请求命令参数

```
{
    "cmd" : "get_encrypt_key",
    "id" : "999999",
    "prime_key" : "Vg0MgpeBCOzzCxbsQ68V2NHd0Zk="
}
```

字段名	说明
prime_key	主密钥字串（加密之后的密码） ● prime_key 的加密方法：由 HMAC-SHA 算法再加上原始的 prime_key，对 signature 进行加密生成数据，再进行 Base64 生成的字符串。Signature 可通过 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt 四个接口获取，需采用最新的 signature

注意：prime_key 的加密与 new_encrypt_key 的加密可借助上面提到的网址得到结果

结果返回

```
{
    "cmd" : "reset_encrypt_key",
    "id" : "999999",
    "signature" : "Rr17TBjPHNYXJR80",
    "encrypt_key" : "OB46x4SOg0ZTXa9BDFGh/i5LAfQ=",
    "state_code" : 200
}
```

字段名	说明
signature	一个十六位的字符串（用于对传输的密码进行加密，每调用一次 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt 四个接口中的任意一个接口, signature 都会变，重置/修改密码和开启加密需采用最新的 signature）
state_code:	状态码 ● 200：成功 ● 401：用户没有权限 ● 400：用户参数设置错误（有可能密码太长或者太短） ● 500：服务器内部错误（内部设置错误）
encrypt_key	**encrypt_key** 的加密方法：使用 AES 128 位算法，使用秘钥 prime_key（原始主密钥），对密码进行加密生成数据，再进行 Base64 生成的字符串

2.7.3 重新设置用户密码：reset_encrypt_key

用户根据主密钥重新设置用户密码：用户在忘记自己设置的密码时可根据主密钥重新设置自己的密码

请求命令参数

```
{
    "cmd" : "reset_encrypt_key",
    "id" : "999999",
    "new_encrypt_key" : "j6NrhO8gWFKU0O8mk8+A/g==",
    "prime_key" : "Vg0MgpeBCOzzCxbsQ68V2NHd0Zk="
}
```

字段名	说明
prime_key	主密钥字串（加密之后的密码） ● prime_key 的加密方法：由 HMAC-SHA 算法再加上原始的 prime_key（主密码），对 signature 进行加密生成数据，再进行 Base64

	生成的字符串。Signature 可通过 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt 四个接口获取，需采用最新的 signature
new_encrypt_key	新设置的用户密码（加密之后的密码） ● new_encrypt_key 的加密方法：使用 AES 128 位算法，使用密码 prime_key（原始主密钥），对新的密码进行加密生成数据，再进行 Base64 生成的字符串

注意：prime_key 的加密与 new_encrypt_key 的加密可借助上面提到的网址得到结果

结果返回

```
{
    "cmd" : "reset_encrypt_key",
    "id" : "999999",
    "signature" : "Rr17TBjPHNYXJR80",
    "state_code" : 200
}
```

字段名	说明
signature	一个十六位的字符串（用于对传输的密码进行加密，每调用一次 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt 四个接口中的任意一个接口, signature 都会变，重置/修改密码和开启加密需采用最新的 signature）
state_code:	状态码 ● 200：成功 ● 401：用户没有权限 ● 400：用户参数设置错误（有可能密码太长或者太短） ● 500：服务器内部错误（内部设置错误）

2.7.4 修改用户密码：change_encrypt_key

将旧的用户密码修改成新的用户密码或者将默认的用户密码修改成新的用户密码

请求命令参数

```
{
    "cmd" : "change_encrypt_key",
    "id" : "999999",
    "encrypt_key" : "OB46x4SOg0ZTXa9BDFGh/i5LAfQ=",
    "new_encrypt_key" : "sZuBTdl+Fy0HjgFJ/yDmSA=="
}
```

字段名	说明
encrypt_key	旧的用户密码（加密后的） ● encrypt _key 的加密方法：使用 HMAC-SHA1 算法对原始用户密码加密(encrypt _key)，加密物料使用 signature，再进行 Base64 生成字符串。 ● Signature 可通过 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt 四个接口获取，需采用最新的 signature
new_encrypt_key	新设置的用户密码（加密之后的密码）

	● new_encrypt_key 的加密方法：使用 AES 128 位算法，使用原本的 encrypt _key（原始用户密码），对新的密码进行加密生成数据，再进行 Base64 生成的字符串

注意：encrypt_key 的加密与 new_encrypt_key 的加密可借助上面提到的网址得到结果

结果返回

```
{
    "cmd" : "change_encrypt_key",
    "signature" : "8GZiU0f8u2slTCsp",
    "id" : "999999",
    "state_code" : 200
}
```

字段名	说明
signature	一个十六位的字符串（用于对传输的密码进行加密，每调用一次 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt 四个接口中的任意一个接口, signature 都会变，重置/修改密码和开启加密需采用最新的 signature）
state_code:	状态码 ● 200：成功 ● 401：用户没有权限 ● 400：用户参数设置错误（有可能密码太长或者太短） ● 500：服务器内部错误（内部设置错误）

如

```
std::string old_key;
std::string encrypt_key;
unsigned char digest_buffer[20] = {0};
HmacSha1((const unsigned char *) old_key.c_str(),old_key.size(),
    (const unsigned char *) signature.c_str(),signature.size(), digest_buffer);
Base64::EncodeFromArray(digest_buffer,    20, &encrypt_key);

std::string new_key;
std::string new_encrypt_key;
AES128_ECB_encrypt((const unsigned char *) old_key.c_str(),
                        (const unsigned char *) new_key.c_str(), new_encrypt_key);
Base64::EncodeFromArray(new_encrypt_key.c_str(),new_encrypt_key.size(), & new_encrypt_key);
```

2.7.5 开启是否加密：enable_encrypt

请求命令参数

注意：m_id ： 0 为不加密，encrypt_key 需传入用户密码（加密后的），而非主密钥

```
{
    "cmd" : "enable_encrypt",
    "encrypt_key" : "9k1t5PhnUXlfgJKgHmcswU3PIp8=",
    "id" : "999999",
    "m_id" : 1
}
```

字段名	说明

encrypt_key	用户密码（加密后的） ● encrypt _key 的加密方法：使用 HMAC-SHA1 算法对原始用户密码加密(encrypt _key)，加密物料使用 signature，再进行 Base64 生成字符串。 ● 。Signature 可通过 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt 四个接口获取，需采用最新的 signature
m_id	加密方式的索引（m_id : 0 为不加密）

注意：encrypt_key 的加密可借助上面提到的网址得到结果

结果返回

```
{
    "cmd" : "enable_encrypt",
    "signature" : "BUqCmKfiKt7dYEOv",
    "id" : "999999",
    "state_code" : 200
}
```

字段名	说明
signature	一个十六位的字符串（用于对传输的密码进行加密，每调用一次 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt 四个接口中的任意一个接口, signature 都会变，重置/修改密码和开启加密需采用最新的 signature）
State_code:	状态码 ● 200：成功 ● 401：用户没有权限 ● 400：用户参数设置错误（m_id 错误） ● 500：服务器内部错误（内部设置错误）

.

2.7.6 设置设备有效时间：device_active_settings

设置启用授权后，超过相机授权时间后，相机不再推送识别结果；

请求命令参数

```
{
    "cmd": "device_active_settings",
    "id" : "123",
    "body":{
        "type":"set_device_active_status",
        "active_status":1,
        "active_time":60,
        "authentication":"asdgzxb"
    }
}
```

回复结果：

```
{
    "body" : {
        "state" : 200,
        "type" : "set_device_active_status"
```

```
    },
    "cmd" : "device_active_settings",
    "id" : "123",
    "state_code" : 200
}
```

字段名	说明
active_status	是否启用相机授权 0 不启用 1 启用 启用授权后，在超过授权时间后，相机不再推送识别结果信息；
active_time	相机授权时间，单位秒；
authentication	加密后再经过 64 位编码的用户密码 使用 SHA1 算法对用户密码加密，加密物料使用签名，再进行 64 位编码后的字符串；

2.7.7 获取设备有效时间：device_active_settings

请求命令参数

```
{
    "cmd": "device_active_settings",
    "id" : "123",
    "body":{
        "type":"get_device_active_status"
    }
}
```

回复结果：

```
{
    "body" : {
        "active_status" : 1,
        "active_time" : 1200,
        "signature" : "cR6dxChrE42JufOQ",
        "state" : 200,
        "type" : "get_device_active_status"
    },
    "cmd" : "device_active_settings",
    "id" : "123",
    "state_code" : 200
}
```

2.7.8 设置用户私有数据：set_user_data

请求命令参数

```
{
    "cmd": "set_user_data",
    "id" : "123",
    "body":{
    "data":"MTMyNDU2Nzg5
```

```
            "
        }
    }
```

回复结果：

```
{
    "cmd" : "set_user_data",
    "id" : "123",
    "state_code" : 200
}
```

字段名	说明
data	需要设置的用户数据，经过 base64 位编码

2.7.9 获取用户私有数据：get_user_data

请求命令参数

```
{
"cmd" : "get_user_data",
"id":"123"
}
```

回复结果：

```
{
    "body" : {
        "data" : "MTMyNDU2Nzg5"
    },
    "cmd" : "get_user_data",
    "id" : "123",
    "state_code" : 200
}
```

字段名	说明
data	需要设置的用户数据，经过 base64 位编码

2.8 设备组网：dg_json_request

设备组网不同于传统的将一台设备的信息提供给用户，需是自动的将一定区域内的所有设备信息提供给用户，以便应用层用户能够更加方便的接入我们的设备系统，更加快速的开发自己的应用。

设备组网功能主要应用在停车场环境，一个停车场里面的所有设备之间相互连接组成一个网络。当有一辆车进入或者离开整个停车场的时候，设备之间进行信息的共享和同步，并且根据这些数据分析出一个最优的结果，然后传输给用户，以方便应用开发，甚至直接收费。

设备组网协议主要分为两个大部分。第一步是关于识别结果的协议。第二部分是关于各种操作组网的协议。

请求命令参数如下格式：

```
{
  "cmd": "dg_json_request",
```

```
    "id": "132156",
    "body": {
          "type": "%s"
          ...//此处为组网请求细分消息中携带的参数，具体如下
      }
}
```

结果返回

0. 处理成功：

```
{
   "cmd" : "dg_json_request",
   "id" : "132156",
   "state_code" : 200,
    "body" : {
   "state" : 200,
   "type" : "get_cdvzid",
         ……// 具体请求具体回复
    }
}
```

1. 格式错误：

```
{
   "cmd" : "dg_json_request",
   "id" : "132156",
    "state_code" : 400
}
```

2. 服务器处理超时错误：

```
{
   "cmd" : "dg_json_request",
   "id" : "132156",
    "state_code" : 408
}
```

3. 服务器内部错误：

```
{
   "cmd" : "dg_json_request",
   "id" : "132156",
    "state_code" : 500
}
```

2.8.1 得到当前设备 vzid：get_cdvzid

我们当前将每一个设备的名称定义为 vzid ， 用来标记这一台设备

请求命令参数

```
{
    "cmd": "dg_json_request",
    "id": "132156",
    "body": {
        "type": "get_cdvzid"
    }
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30

body	需要传递给组网模块的数据
type	操作组网模块的命令 ● get_cdvzid 获取当前设备 vzid

结果返回

```
{
    "body" : {
        "state" : 200,
        "type" : "get_cdvzid",
        "vzid" : {
            "enable_group" : true,
            "ip_addr" : "192.168.7.22",
            "name" : "bHk=&MTkyLjE2OC43LjIy#",
            "sn" : "56a8872e-2c5f7863",
            "type" : "input"
        }
    },
    "cmd" : "dg_json_request",
    "id" : "132156",
    "state_code" : 200
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
body	组网模块处理后返回的数据
state_code	状态码

Body 内的字段说明：

字段名	说明
type	操作组网模块的命令 get_cdvzid 获取当前设备 vzid
vzid	设备的名称，vzid 说明如下
state	状态码

vzid 字段说明：

字段名	说明
enable_group	开启组网功能 ● true 开启 ● false 关闭
ip_addr	设备 ip
name	设备名字
sn	设备序列号
type	设备类型 ● input 入口设备 ● output 出口设备

2.8.2 得到当前设备名称：get_current_device_name

请求命令参数

```
{
    "cmd": "dg_json_request",
    "id": "132156",
    "body": {
```

```
        "type": "get_current_device_name"
    }
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30
body	需要传递给组网模块的数据

Body 内的字段说明：

字段名	说明
type	操作组网模块的命令 ● **get_current_device_name** 得到当前设备名称

结果返回

返回结果为当前设备 vzid 的信息，和 2.8.1 返回结果一致，请参考；

2.8.3 得到在线设备信息，不含自己：get_ovzid

请求命令参数

```
{
    "cmd": "dg_json_request",
    "id": "132156",
    "body": {
        "type": "get_ovzid"
    }
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30
body	需要传递给组网模块的数据

Body 内的字段说明：

字段名	说明
type	操作组网模块的命令 ● get_ovzid 得到所有已连接的设备信息

结果返回

```
{
    "body": {
        "state": 200,
        "type": "get_ovzid",
        "vzids": [
            {
                "enable_group": true,
                "ip_addr": "192.168.2.29",
                "name": "2.29 TEST",
                "sn": "0f692718-95bc0185",
                "type": "output"
            },
            {
                "enable_group": true,
                "ip_addr": "192.168.2.77",
                "name": "2.77 TEST",
```

```
                "sn": "ef69b450-21bf3bff",
                "type": "input"
            },
            {
                "enable_group": true,
                "ip_addr": "192.168.2.79",
                "name": "2.79 TEST",
                "sn": "c9455af2-68daf655",
                "type": "output"
             },
             {
                "enable_group": true,
                "ip_addr": "192.168.2.9",
                "name": "2.9 TEST",
                "sn": "037de919-76c16e36",
                "type": "output"
             }
            ]
        },
    "cmd": "dg_json_request",
    "id": "132156",
    "state_code": 200
}
```


字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
body	组网模块处理后返回的数据
state_code	状态码

Body 内的字段说明：

字段名	说明
type	操作组网模块的命令 ● get_ovzid 得到所有已连接的设备信息
vzid	设备信息列表（由多条设备信息组成），一条设备信息，即 vzid，vzid 上面已经提到过了

2.8.4 得到在线设备信息，含自己：online_devices

请求命令参数

```
{
    "cmd": "dg_json_request",
    "id": "132156",
    "body": {
        "type": "online_devices"
    }
}
```

字段名	说明
type	操作组网模块的命令 ● **online_devices** 得到所有已连接的设备信息

结果返回：


```
    {
        "cmd" : "dg_json_request",
        "id" : "132156",
        "state_code" : 200,
         "body" :
```

```
                {
                  "state" : 200,
                  “type" : "online_devices",
                  "online_devices" : [
                    {
                     "enable_group" : true,
                     "ip_addr" : "192.168.7.22",
                     "name" : "bHk=&MTkyLjE2OC43LjIy#",
                     "sn" : "56a8872e-2c5f7863",
                     "type" : "input"
                    },
                    {
                      "enable_group" : true,
                      "ip_addr" : "192.168.7.22",
                      "name" : "bHk=&MTkyLjE2OC43LjIy#",
                      "sn" : "56a8872e-2c5f7863",
                      "type" : "input"
                    }
                  ]
            }
      }
```

字段名	说明
type	操作组网模块的命令 ● **online_devices** 得到所有已连接的设备信息 ● **online_devices 中的数据结构与上面 vzids 的数据一致，请参考**

2.8.5 得到所有连接设备信息：get_avzid

请求命令参数

```
{
    "cmd": "dg_json_request",
    "id": "132156",
    "body": {
        "type": "get_avzid"
    }
}
```

结果返回：

结果返回以及字段信息和上面 2.8.3 一致，请参考；

2.8.6 得到当前组网内所有设备信息：get_agdi

请求命令参数

```
{
    "cmd": "dg_json_request",
    "id": "132156",
    "body": {
        "type": "get_agdi"
    }
}
```

字段名	说明
type	操作组网模块的命令 ● **get_agdi** 得到当前组网内所有设备信息

结果返回：

结果返回以及字段信息和上面 2.8.3 一致，请参考；

2.8.7 得到当前设备记录 size：current_records_size

请求命令参数

```
{
   "cmd": "dg_json_request",
   "id": "132156",
    "body": {
              "type": "current_records_size"
    }
}
```

字段名	说明
type	操作组网模块的命令 ● **current_records_size** 得到当前设备记录的 size

结果返回：

1. 入口设备：

```
{
   “body” : {
        “state” : 200,
        “type” : “current_records_size”,
        “input_records_size”:1024,
        “vzid” : {
                  “enable_group” : true,
                  “ip_addr” : “192.168.7.22”,
                  “name” : “bHk=&MTkyLjE2OC43LjIy#”,
                  “sn” : “56a8872e-2c5f7863”,
                  “type” : “input”
                  }
    },
    “cmd” : “dg_json_request”,
    “id” : “132156”,
    “state_code” : 200
}
```

2. 出口设备：

```
{
   “body” : {
        “state” : 200,
        “type” : “current_records_size”,
        “output_records_size”:1024,
        “vzid” : {
                  “enable_group” : true,
                  “ip_addr” : “192.168.7.22”,
                  “name” : “bHk=&MtkyLjE2OC43LjIy#”,
                  “sn” : “56a8872e-2c5f7863”,
                  “type” : “output”
                  }
    },
    “cmd” : “dg_json_request”,
    “id” : “132156”,
    “state_code” : 200
}
```

字段名	说明
output_records_size	● 出口设备记录的 size
input_records_size	● 入口设备记录的 size
Vzid	● 同时返回当前设备 vzid 信息；vzid 信息参考前面；

2.8.8 得到入口设备记录：records_sparate_input

由于设备记录有很多，一般情况下很难一次性传输出来，所以，在得到设备记录的时候必须指定一个范围。在最开始的时候如果不知道记录有
多少条，可以随便指定一个可以传输的范围，32 最好。然后在回复的消息中，会告诉当前设备有多少条记录。这样的分页机制，让用户可以更好的开发自己的应用

请求命令参数

```
{
    "body" : {
        "input_begin_pos" : 0,
        "input_end_pos" : 8,
        "type" : "records_sparate_input"
    },
    "cmd" : "dg_json_request",
    "id" : "123456789"
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30
body	需要传递给组网模块的数据
type	请求回应消息

Body 内的字段说明：

字段名	说明
type	操作组网模块的命令 ● **records_sparate_input** 获取入口设备记录
input_begin_pos	获取记录的开始位置
input_end_pos	获取记录的结束位置

结果返回

```
{
    "body" : {
        "input_begin_pos" : 0,
        "input_end_pos" : 8,
        "input_records_size" : 10213,
        "records" : [
            {
                "enter_ip" : "192.168.5.13",
                "enter_name"                                                                                    :
"6L2m5bqTMzA=&55u46L6FNg==&MTkyLjE2OC41LjEz#dHJlZURlbW9fNQ==",
                "enter_time" : 1494049723,
                "plate" : "渝 AX4462",
                "state" : 5
            },
            {
                "enter_ip" : "192.168.5.13",
                "enter_name"                                                                                    :
"6L2m5bqTMzA=&55u46L6FNg==&MTkyLjE2OC41LjEz#dHJlZURlbW9fNQ==",
                "enter_time" : 1494049731,
```

```
            "plate" : "琼 ECVWJB",
            "state" : 6
        },
        {
            "enter_ip" : "192.168.5.13",
            "enter_name" :
"6L2m5bqTMzA=&55u46L6FNg==&MTkyLjE2OC41LjEz#dHJlZURlbW9fNQ==",
            "enter_time" : 1494049739,
            "plate" : "豫 JDZBHF",
            "state" : 6
        },
        {
            "enter_ip" : "192.168.5.13",
            "enter_name" :
"6L2m5bqTMzA=&55u46L6FNg==&MTkyLjE2OC41LjEz#dHJlZURlbW9fNQ==",
            "enter_time" : 1494049754,
            "plate" : "冀 SCNGRR",
            "state" : 6
        },
        {
            "enter_ip" : "192.168.5.13",
            "enter_name" :
"6L2m5bqTMzA=&55u46L6FNg==&MTkyLjE2OC41LjEz#dHJlZURlbW9fNQ==",
            "enter_time" : 1494049762,
            "plate" : "新 GMKA3M",
            "state" : 6
        },
        {
            "enter_ip" : "192.168.5.13",
            "enter_name" :
"6L2m5bqTMzA=&55u46L6FNg==&MTkyLjE2OC41LjEz#dHJlZURlbW9fNQ==",
            "enter_time" : 1494049770,
            "plate" : "赣 QFWXF5",
            "state" : 6
        },
        {
            "enter_ip" : "192.168.5.13",
            "enter_name" :
"6L2m5bqTMzA=&55u46L6FNg==&MTkyLjE2OC41LjEz#dHJlZURlbW9fNQ==",
            "enter_time" : 1494049778,
            "plate" : "豫 X4JK1G",
            "state" : 6
        },
        {
            "enter_ip" : "192.168.5.13",
            "enter_name" :
"6L2m5bqTMzA=&55u46L6FNg==&MTkyLjE2OC41LjEz#dHJlZURlbW9fNQ==",
            "enter_time" : 1494049784,
            "plate" : "豫 QWY8KH",
            "state" : 6
        },
        {
            "enter_ip" : "192.168.5.13",
            "enter_name" :
"6L2m5bqTMzA=&55u46L6FNg==&MTkyLjE2OC41LjEz#dHJlZURlbW9fNQ==",
            "enter_time" : 1494049793,
            "plate" : "沪 H1JLJ0",
            "state" : 6
        }
    ],
    "state" : 200,
    "type" : "records_sparate_input",
    "vzid" : {
```

```
            "enable_group" : true,
            "ip_addr" : "192.168.7.22",
            "name" : "bHk=&MTkyLjE2OC43LjIy#",
            "sn" : "56a8872e-2c5f7863",
            "type" : "input"
        }
    },
    "cmd" : "dg_json_request",
    "id" : "123456789",
    "state_code" : 200
}
```


字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
body	组网模块处理后返回的数据
state_code	状态码

Body 内的字段说明：

字段名	说明
input_begin_pos	获取记录的开始位置
input_end_pos	获取记录的结束位置
input_records_size	入口设备记录总条数
records	入口设备记录的列表（由多条入口记录组成）
state	状态码
type	操作组网模块的命令 ● records_sparate_input 获取入口设备记录
vzid	设备的名称，vzid 说明如下

records 内的字段说明:

字段名	说明
enter_ip	ip 地址
enter_name	设备名称
enter_time	识别时间点
plate	车牌号码（汉字为 GB2312 编码）
state	设备的名称，vzid 说明如下

vzid 字段说明：

字段名	说明
enable_group	开启组网功能 ● true 开启 ● false 关闭
ip_addr	设备 ip
name	设备名字
sn	设备序列号
type	设备类型

	• input 入口设备 • output 出口设备

2.8.9 得到出口设备记录：records_sparate_output

请求命令参数

```
{
    "body" : {
        "output_records_begin_pos" : 0,
        "output_records_end_pos" : 8,
        "type" : "records_sparate_output"
    },
    "cmd" : "dg_json_request",
    "id" : "123456789"
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30
body	需要传递给组网模块的数据

Body 内的字段说明：

字段名	说明
type	操作组网模块的命令 • **records_sparate_outpu** 获取出口设备记录
output_records_begin_pos	获取记录的开始位置
output_records_end_pos	获取记录的结束位置

结果返回

```
{
    "body" : {
        "output_records_begin_pos" : 0,
        "output_records_end_pos" : 8,
        "output_records_size" : 4549,
        "records" : [
            {
                "enter_ip" : "192.168.4.79",
                "enter_name" : "5ryr5q2l5LqR56uv&MTkyLjE2OC40Ljc5#",
                "enter_time" : 1465496977,
                "leave_ip" : "192.168.7.22",
                "leave_name" : "bHk=&MTkyLjE2OC43LjIy#",
                "leave_time" : 1465496992,
                "plate" : "粤 Z72D2 港",
                "state" : 0
            },
            {
                "enter_ip" : "192.168.4.97",
                "enter_name" : "5ryr5q2l5LqR56uv&MTkyLjE2OC40Ljk3#",
                "enter_time" : 1465497011,
                "leave_ip" : "192.168.7.22",
                "leave_name" : "bHk=&MTkyLjE2OC43LjIy#",
                "leave_time" : 1465497011,
                "plate" : "粤 Z72D2 港",
                "state" : 3
            },
            {
                "enter_ip" : "192.168.4.97",
```

```
        "enter_name" : "5ryr5q2l5LqR56uv&MTkyLjE2OC40Ljk3#",
        "enter_time" : 1465497051,
        "leave_ip" : "192.168.7.22",
        "leave_name" : "bHk=&MTkyLjE2OC43LjIy#",
        "leave_time" : 1465497051,
        "plate" : "粤 Z72D2 港",
        "state" : 3
    },
    {
        "enter_ip" : "192.168.4.97",
        "enter_name" : "5ryr5q2l5LqR56uv&MTkyLjE2OC40Ljk3#",
        "enter_time" : 1468906933,
        "leave_ip" : "192.168.7.22",
        "leave_name" : "bHk=&MTkyLjE2OC43LjIy#",
        "leave_time" : 1468906933,
        "plate" : "粤 Z72D2 港",
        "state" : 3
    },
    {
        "enter_ip" : "192.168.4.97",
        "enter_name" : "5ryr5q2l5LqR56uv&MTkyLjE2OC40Ljk3#",
        "enter_time" : 1468906941,
        "leave_ip" : "192.168.7.22",
        "leave_name" : "bHk=&MTkyLjE2OC43LjIy#",
        "leave_time" : 1468906941,
        "plate" : "粤 Z72D2 港",
        "state" : 3
    },
    {
        "enter_ip" : "192.168.4.79",
        "enter_name" : "5ryr5q2l5LqR56uv&MTkyLjE2OC40Ljc5#",
        "enter_time" : 1468906962,
        "leave_ip" : "192.168.7.22",
        "leave_name" : "bHk=&MTkyLjE2OC43LjIy#",
        "leave_time" : 1468906967,
        "plate" : "粤 Z72D2 港",
        "state" : 0
    },
    {
        "enter_ip" : "192.168.4.97",
        "enter_name" : "5ryr5q2l5LqR56uv&MTkyLjE2OC40Ljk3#",
        "enter_time" : 1468907000,
        "leave_ip" : "192.168.7.22",
        "leave_name" : "bHk=&MTkyLjE2OC43LjIy#",
        "leave_time" : 1468907000,
        "plate" : "粤 Z72D2 港",
        "state" : 3
    },
    {
        "enter_ip" : "192.168.4.79",
        "enter_name" : "5ryr5q2l5LqR56uv&MTkyLjE2OC40Ljc5#",
        "enter_time" : 1468907153,
        "leave_ip" : "192.168.7.22",
        "leave_name" : "bHk=&MTkyLjE2OC43LjIy#",
        "leave_time" : 1468907367,
        "plate" : "粤 Z72D2 港",
        "state" : 0
    },
    {
        "enter_ip" : "192.168.4.97",
        "enter_name" : "5ryr5q2l5LqR56uv&MTkyLjE2OC40Ljk3#",
        "enter_time" : 1468907448,
        "leave_ip" : "192.168.7.22",
```

```
                "leave_name" : "bHk=&MTkyLjE2OC43LjIy#",
                "leave_time" : 1468907448,
                "plate" : "粤 Z0V89 港",
                "state" : 3
            }
        ],
        "state" : 200,
        "type" : "records_sparate_output",
        "vzid" : {
            "enable_group" : true,
            "ip_addr" : "192.168.7.22",
            "name" : "bHk=&MTkyLjE2OC43LjIy#",
            "sn" : "56a8872e-2c5f7863",
            "type" : "output"
        }
    },
    "cmd" : "dg_json_request",
    "id" : "123456789",
    "state_code" : 200
}
```


字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
body	组网模块处理后返回的数据
state_code	状态码

Body 内的字段说明：

字段名	说明
output_records_begin_pos	获取记录的开始位置
output_records_end_pos	获取记录的结束位置
output_records_size	出口设备记录总条数
records	出口设备记录的列表（由多条出口记录组成，一条出口记录由此车牌的入口记录信息+此车牌的出口记录信息组成）
state	状态码
type	操作组网模块的命令 ● **records_sparate_output** 获取出口设备记录
vzid	设备的名称

vzid 字段说明：

字段名	说明
enable_group	开启组网功能 ● true 开启 ● false 关闭
ip_addr	设备 ip
name	设备名字
sn	设备序列号
type	设备类型 ● input 入口设备 ● output 出口设备

2.8.10 使能设备组网：enable_devicegroup

请求命令参数

```
{
    "cmd": "dg_json_request",
     "id": "132156",
     "body": {
     "type": "enable_devicegroup",
     "vzid" : {
        "enable_group" : true,
        "ip_addr" : "192.168.7.22",
        "name" : "bHk=&MTkyLjE2OC43LjIy#",
        "sn" : "56a8872e-2c5f7863",
        "type" : "input"
         }
    }
}
```

Body 内的字段说明：

字段名	说明
type	操作组网模块的命令 ● **enable_devicegroup 使能设备组网**
vzid	需要使能的设备的 vzid
	Vzid 中的字段意思，参考 2.8.1 中关于 vzid 的描述

结果返回

```
{
                                "cmd" : "dg_json_request",
                                "id" : "132156",
                                "state_code" : 200
                                "body" :
                                {
                                    "state" : 200,
                                    "type" : "enable_devicegroup"
                                }
}
```

2.8.11 设置设备组网类型及参数：set_device_type_enable

功能和 2.8.10 一致

请求命令参数

```
{
    "cmd": "dg_json_request",
     "id": "132156",
     "body": {
     "type": "set_device_type_enable",
     "vzid" : {
        "enable_group" : true,
        "ip_addr" : "192.168.7.22",
        "name" : "bHk=&MTkyLjE2OC43LjIy#",
        "sn" : "56a8872e-2c5f7863",
        "type" : "input"
         }
    }
```

```
}
```

Body 内的字段说明：

字段名	说明
type	操作组网模块的命令 ● set_device_type_enable **设置设备组网**
vzid	需要使能的设备的 vzid
	Vzid 中的字段意思，参考 2.8.1 中关于 vzid 的描述

结果返回

```
{
                    "cmd" : "dg_json_request",
                    "id" : "132156",
                    "state_code" : 200
                    "body" :
                    {
                        "state" : 200,
                        "type" : "set_device_type_enable"
                    }
}
```

2.8.12 查找车牌信息：search_plate

请求命令参数

```
{
                    "cmd": "dg_json_request",
                    "id": "132156",
                    "body": {
                                "type": "search_plate",
                                "plate":"川 A12345",
                                "record_type": "input",
                                "record_state": 5
                            }
}
```

字段名	说明
type	操作组网模块的命令 ● **search_plate 查找车牌信息**
Plate	车牌号
record_type	记录类型 :入场设备; ● input ：入场 ● output:出场 select 白名单查询
record_state	记录状态

结果返回

```
{
                    "cmd" : "dg_json_request",
                    "id" : "132156",
                    "state_code" : 200,
                    "body" :
```

```
{
    "state" : 200,
    "type" : "search_plate",
    "search_result":
    {
        "record_type":"output",
        "output_record":
        {
            "ivs_result_param":
            {
                "bright" : 0,
                "carBright" : 0,
                "carColor" : 0,
                "colorType" : 1,
                "colorValue" : 0,
                "confidence" : 99,
                "direction" : 0,
                "license" : "青 PTW3Z3",
                "location" : {
                    "RECT" : {
                        "bottom" : 392,
                        "left" : 690,
                        "right" : 834,
                        "top" : 350
                    }
                },
                "timeStamp" : {
                    "Timeval" : {
                        "sec" : 1458882234,
                        "usec" : 921325
                    }
                },
                "timeUsed" : 0,
                "triggerType" : 1,
                "type" : 1
                },
                "active_id" : 0,
                "clipImgSize" : 1103,
                "cmd" : "ivs_result",
                "fullImgSize" : 51566,
                "id" : 0,
                "imageformat" : "jpg",
                "timeString" : "2016-03-25 13:03:54"
            },
            "device_name":"asgadg",
            "state":2
        },
        "output_record":
        {
            "ivs_result_param":
            {
                "bright" : 0,
                "carBright" : 0,
                "carColor" : 0,
                "colorType" : 1,
                "colorValue" : 0,
                "confidence" : 99,
                "direction" : 0,
                "license" : "青 PTW3Z3",
                "location" : {
                    "RECT" : {
                        "bottom" : 392,
                        "left" : 690,
```

```
                                        "right" : 834,
                                        "top" : 350
                                    }
                                },
                                "timeStamp" : {
                                    "Timeval" : {
                                        "sec" : 1458882234,
                                        "usec" : 921325
                                    }
                                },
                                "timeUsed" : 0,
                                "triggerType" : 1,
                                "type" : 1
                                },
                                "active_id" : 0,
                                "clipImgSize" : 1103,
                                "cmd" : "ivs_result",
                                "fullImgSize" : 51566,
                                "id" : 0,
                                "imageformat" : "jpg",
                                "timeString" : "2016-03-25 13:03:54"

                            },
                            "device_name":"asgadg",
                            "state":2
                        }
                    },
                    {
                        "record_type":"input",
                        "input_record":
                        {
                            "ivs_result_param":
                            {
                                // 参考识别结果参数
                            },
                            "device_name":"asgadg",
                            "state":2
                        }
                    }
                }
}
```

字段名	说明
type	操作组网模块的命令 ● search_plate 查找车牌信息

search_result 中字段含义：

字段名	说明
record_type	● 当前记录是出场还是入场记录
output_record	● 出场记录
input_record	● 入场记录
ivs_result_param	● 车辆信息，结构体
State	● 当前车辆状态
device_name	● 设备名称

ivs_result_param 中字段含义：
参考 2.2.1 中车牌识别结果中字段含义

2.8.13 获取组网匹配模式：get_device_match_mode

请求命令参数

```
{
                        "cmd": "dg_json_request",
                        "id": "132156",
                        "body": {
                                    "type": "get_device_match_mode"
                                }
}
```

结果返回

```
{
                            "cmd" : "dg_json_request",
                            "id" : "132156",
                            "state_code" : 200
                            "body" :
                            {
                                    "state" : 200,
                                    "type" : " get_device_match_mode ",
                                    "mode":" fuzzy_mode "
                             }
}
```

字段名	说明
mode	组网匹配模式： fuzzy_mode：模糊匹配 exact_mode：精确匹配

2.8.14 设置组网匹配模式: set _device_match_mode

请求命令参数

```
{
                            "cmd" : "dg_json_request",
                            "id" : "132156",
                            "body" :
                            {
                                  "type" : "set_device_match_mode",
                                   "mode":"fuzzy_mode"
                             }
}
```

结果返回

```
{
                            "cmd" : "dg_json_request",
                            "id" : "132156",
                            "state_code" : 200
                            "body" :
                            {
                                    "state" : 200,
                                    "type" : "set_device_match_mode"
                             }
                        }
```

字段名	说明

mode	组网匹配模式： **fuzzy_mode：模糊匹配** exact_mode：精确匹配

2.8.15 得到组网共享 IO：get_group_shared_io

请求命令参数

```
{
                "cmd": "dg_json_request",
                "id": "132156",
                "body": {
                        "type": "get_group_shared_io"
                }
}
```

结果返回

```
{
                "cmd" : "dg_json_request",
                "id" : "132156",
                "state_code" : 200
                "body" :
                {
                        "state" : 200,
                        "type" : "get_group_shared_io",
                        "value": 1
                 }
}
```

字段名	说明
value	当前组网共享的 IO 的值

2.8.16 设置组网共享 IO：set_group_shared_io

请求命令参数

```
{
                "cmd" : "dg_json_request",
                "id" : "132156",
                "body" :
                {
                        "type" : "set_group_shared_io",
                        "value": 1
                 }
}
```

结果返回

```
{
                "cmd" : "dg_json_request",
                "id" : "132156",
                "state_code" : 200
                "body" :
                {
                        "state" : 200,
                        "type" : "set_group_shared_io"
                 }
}
```

2.8.17 组网识别结果消息 enable_dg_result

识别结果消息分成了两种类型一种类型是入口设备的识别结果消息，一种是出口设备的识别结果消息。用户首先需要开启允许接收组网识别结果消息，未来，如果有车牌识别消息到来，用户才会接收到出/入口消息

请求命令参数

```
{
    "cmd":"enable_dg_result",
    "enable":true,
    "id":"123456"
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30
enable	是否允许接收组网识别消息 ● true 允许接收组网识别消息 ● false 不接收组网识别消息

结果返回

```
{
    "cmd" : "enable_dg_result",
    "id" : "123456",
    "state_code" : 200
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
state_code	状态码

当开启接收组网识别结果后，当组网设备有识别到结果，会主动推送给用户，推送信息如下：

出口消息

```
{
"body": {
"record_type": "output",
"input_record": {
"device_name": {
"enable_group": true,
"ip_addr": "192.168.4.180",
"name": "4.180Leave",
"sn": "7667931c-7fa08a25",
"type": "output"
                },
"ivs_result_param": {
"bright": 0,
"car_bright": 0,
"car_color": 0,
"confidence": 98,
```

```
"direction": 4,
"fragment_path": "/tmp/app/html/snapshot/lpr/patch_tri_snap_4.jpg",
"id": 0,
"image_path": "/tmp/app/html/snapshot/lpr/tri_snap_4.jpg",
"image_sd_path": "/media/mmcblk0p1/VzIPCCap/2015_11_12/1534433501_陕 D4FY61.jpg",
"location": {
"bottom": 309,
"left": 965,
"right": 1150,
"top": 268
                        },
"n_time": 0,
"plate": "陕 D4FY61",
"plate_color": 1,
"plate_type" : 0,
"related_plate" : "陕 D4FY61",
"timeval" : {
                    "decday" : 5,
                    "dechour" : 15,
                    "decmin" : 26,
                    "decmon" : 5,
                    "decsec" : 14,
                    "decyear" : 2017,
                    "tv_sec" : 1493969174,
                    "tv_usec" : 416656
                },
"trig_type": 1
                    },
"state": 0
            },
"output_record": {
"device_name": {
"enable_group": true,
"ip_addr": "192.168.4.180",
"name": "4.180Leave",
"sn": "7667931c-7fa08a25",
"type": "output"
                    },
"ivs_result_param": {
"bright": 0,
"car_bright": 0,
"car_color": 0,
"confidence": 98,
"direction": 4,
"fragment_path": "/tmp/app/html/snapshot/lpr/patch_tri_snap_4.jpg",
"id": 0,
"image_path": "/tmp/app/html/snapshot/lpr/tri_snap_4.jpg",
"image_sd_path": "/media/mmcblk0p1/VzIPCCap/2015_11_12/1534433501_陕 D4FY61.jpg",
"location": {
"bottom": 309,
"left": 965,
"right": 1150,
"top": 268
                        },
"n_time": 0,
"plate": "陕 D4FY61",
"plate_color": 1,
"plate_type" : 0,
"related_plate" : "陕 D4FY61",
"timeval" : {
                    "decday" : 5,
                    "dechour" : 15,
                    "decmin" : 26,
```

```
                "decmon" : 5,
                "decsec" : 14,
                "decyear" : 2017,
                "tv_sec" : 1493969174,
                "tv_usec" : 416656
            },
"trig_type": 1
                },
"state": 0
            }
        },
"cmd": "dg_plateinfo_result",
"id": "132156",
"state_code": 200
}
```


字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
state_code	状态码
cmd	操作命令 ● dg_plateinfo_result 组网识别消息结果

Body 内的字段说明：

字段名	说明
record_type	消息类型 ● output 出口消息 ● input 入口消息
input_record	上面已经提到过了，请自行查找
output_record	上面已经提到过了，请自行查找

入口消息


```
{
"body": {
"record_type": "input",
"input_record": {
"device_name": {
"enable_group": true,
"ip_addr": "192.168.4.157",
"name": "4.157Enter",
"sn": "0ba76bc6-e26d52eb",
"type": "input"
                },
"ivs_result_param": {
"bright": 0,
"car_bright": 0,
"car_color": 0,
"confidence": 99,
"direction": 0,
"fragment_path": "/tmp/app/html/snapshot/lpr/patch_tri_snap_16.jpg",
"id": 0,
"image_path": "/tmp/app/html/snapshot/lpr/tri_snap_16.jpg",
"image_sd_path": "/media/mmcblk0p1/VzIPCCap/2015_11_12/1533300901_陕 A8T00 学.jpg",
"location": {
"bottom": 590,
"left": 587,
```

```
"right": 696,
"top": 571
                    },
"n_time": 0,
"plate": "陕 A8T00 学",
"plate_color": 13,
"plate_type" : 0,
"related_plate" : "陕 A8T00 学",
"timeval" : {
                "decday" : 5,
                "dechour" : 15,
                "decmin" : 26,
                "decmon" : 5,
                "decsec" : 14,
                "decyear" : 2017,
                "tv_sec" : 1493969174,
                "tv_usec" : 416656
            },
"trig_type": 1
                },
"state": 1
            }
     },
"cmd": "dg_plateinfo_result",
"id": "132156",
"state_code": 200
}
```


字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
state_code	状态码
cmd	操作命令 dg_plateinfo_result 组网识别消息结果

Body 内的字段说明：

字段名	说明
record_type	消息类型 output 出口消息 input 入口消息
input_record	上面已经提到过了，请自行查找

2.8.18 清除组网数据：reset_database

请求命令参数


```
{
    "body" : {
        "type" : "reset_database"
    },
    "cmd" : "dg_json_request",
    "id" : "132156"
}
```


字段名	说明
id	序列字符串（唯一），字符串长度 < 30
body	需要传递给组网模块的数据

Body 内的字段说明：

字段名	说明
type	操作组网模块的命令 ● reset_database 清除组网数据

结果返回

```
{
    "body" : {
        "state" : 200,
        "type" : "reset_database"
    },
    "cmd" : "dg_json_request",
    "id" : "132156",
    "state_code" : 200
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
body	组网模块处理后返回的数据
state_code	状态码

Body 内的字段说明:

字段名	说明
type	操作组网模块的命令 reset_database 清除组网数据
state	状态码

2.8.19 得到当前组网内部所有设备配置：get_group_cfg

请求命令参数

```
{
    "body" : {
        "type" : "get_group_cfg"
    },
    "cmd" : "dg_json_request",
    "id" : "132156"
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30
body	需要传递给组网模块的数据

Body 内的字段说明：

字段名	说明
type	操作组网模块的命令 ● get_group_cfg 得到当前组网内部所有设备配置

结果返回


```
{
    "body" : {
        "group_cfg" : {
            "group_vzids" : [
                {
                    "connect_status" : 5,
                    "enable_group" : true,
                    "ip_addr" : "192.168.105.39",
                    "name" : "",
                    "sn" : "018cd419-5e0702a4",
                    "type" : "unkown"
                },
                {
                    "connect_status" : 5,
                    "enable_group" : true,
                    "ip_addr" : "192.168.103.16",
                    "name" : "",
                    "sn" : "0a0248b7-f9121dba",
                    "type" : "unkown"
                },
                {
                    "connect_status" : 5,
                    "enable_group" : true,
                    "ip_addr" : "192.168.8.94",
                    "name" : "",
                    "sn" : "10391362-00e22a20",
                    "type" : "unkown"
                },
                {
                    "connect_status" : 5,
                    "enable_group" : true,
                    "ip_addr" : "192.168.4.186",
                    "name" : "6L2m5bqTNQ==&55u46L6FNTk=&MTkyLjE2OC41LjEx#dHJlZURlbW9fMTU=",
                    "sn" : "169d39b6-34645de7",
                    "type" : "input"
                },
                {
                    "connect_status" : 5,
                    "enable_group" : true,
                    "ip_addr" : "192.168.4.132",
                    "name" : "",
                    "sn" : "3acd054f-6b3c0af3",
                    "type" : "input"
                },
                {
                    "connect_status" : 5,
                    "enable_group" : true,
                    "ip_addr" : "192.168.1.7",
                    "name" : "",
                    "sn" : "702cdcaa-597a919f",
                    "type" : "unkown"
                },
                {
                    "connect_status" : 5,
                    "enable_group" : true,
                    "ip_addr" : "192.168.102.40",
                    "name" : "",
                    "sn" : "783f0ea9-90485466",
                    "type" : "unkown"
                },
```

```
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.22.71",
                "name" : "",
                "sn" : "7a827083-62ad93cd",
                "type" : "input"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.3.3",
                "name" : "",
                "sn" : "7aea8606-8d88dd06",
                "type" : "unkown"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.102.4",
                "name" : "",
                "sn" : "a9132b96-4af3b688",
                "type" : "unkown"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.4.137",
                "name" : "",
                "sn" : "abf20a38-23f0b33d",
                "type" : "unkown"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.1.187",
                "name" : "",
                "sn" : "abf775b8-bf50d0b2",
                "type" : "unkown"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.105.38",
                "name" : "",
                "sn" : "af52ed95-8deaeadb",
                "type" : "unkown"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.102.41",
                "name"                                                                                    :
"ODEyNy0t5Yir5Yqo&55u46L6FMQ==&MTkyLjE2OC4xMDIuNDE=#dHJlZURlbW9fNg==",
                "sn" : "b189c806-faec3304",
                "type" : "output"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.12.251",
                "name" : "",
                "sn" : "b1a7806e-326e00a2",
```

```
                "type" : "unkown"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.20.10",
                "name" : "",
                "sn" : "ed2045b2-30acf3a2",
                "type" : "unkown"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.102.56",
                "name" : "",
                "sn" : "f0565a83-fd3be473",
                "type" : "unkown"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.105.22",
                "name" : "",
                "sn" : "fcf2e551-a44c1cce",
                "type" : "unkown"
            },
            {
                "connect_status" : 5,
                "enable_group" : true,
                "ip_addr" : "192.168.105.14",
                "name" : "",
                "sn" : "ffedb27b-cd76bd51",
                "type" : "unkown"
            }
        ]
    },
    "state" : 200,
    "type" : "get_group_cfg",
    "vzid" : {
        "enable_group" : true,
        "ip_addr" : "192.168.7.22",
        "name" : "bHk=&MTkyLjE2OC43LjIy#",
        "sn" : "56a8872e-2c5f7863",
        "type" : "output"
    }
  },
  "cmd" : "dg_json_request",
  "id" : "132156",
  "state_code" : 200,
  "type" : null
}
```


字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
body	组网模块处理后返回的数据
state_code	状态码

body 内的字段说明：

字段名	说明

group_cfg	组网内所有设备配置
state	状态码
type	操作组网模块的命令 ● get_group_cfg 得到当前组网内部所有设备配置
vzid	设备的名称

group_cfg 内的字段说明：

字段名	说明
group_vzids	组网内所有设备配置信息)(由多条设备配置信息组成)

group_vzids 内的字段说明：

字段名	说明
connect_status	连接状态
enable_group	开启组网功能 ● true 开启 ● false 关闭
ip_addr	IP 地址
name	设备的名称
sn	设备序列号
type	设备类型 ● input 入口设备 ● output 出口设备

vzid 字段说明：

字段名	说明
enable_group	开启组网功能 ● true 开启 ● false 关闭
ip_addr	设备 ip
name	设备名字
sn	设备序列号
type	设备类型 ● input 入口设备 ● output 出口设备

2.8.20 设置当前组网内部所有设备配置：set_group_cfg

请求命令参数

```
{
                "cmd": "dg_json_request",
                "id": "132156",
                "body": {
                        "type": "set_group_cfg",
                        "vzid" :
                        {
                                "enable_group" : true,
                                "ip_addr" : "192.168.7.22",
```

```
                                        "name" : "bHk=&MTkyLjE2OC43LjIy#",
                                        "sn" : "56a8872e-2c5f7863",
                                        "type" : "input"
                                    },
                                    "group_cfg":
                                    [{
                                        "enable_group" : true,
                                        "ip_addr" : "192.168.1.2",
                                        "name" : "bHk=&asdg4231#",
                                        "sn" : "1238872e-2c5fasdv",
                                        "type" : "input",
                                        "connect_status": 2
                                    },
                                    {
                                        "enable_group" : true,
                                        "ip_addr" : "192.168.22.12",
                                        "name" : "bHk=&456sdg13#",
                                        "sn" : "asxc887e-2cafasdv",
                                        "type" : "input",
                                        "connect_status": 2
                                    }]
                                }
}
```

结果返回

```
{
                                "cmd" : "dg_json_request",
                                "id" : "132156",
                                "state_code" : 200
                                "body" :
                                {
                                    "state" : 200,
                                    "type" : "set_group_cfg"
                                 }
}
```

2.8.21 根据 ID 获取组网图片：get_img_by_id

请求命令参数

```
{
    "body" : {
        "image_id" : 79509,
        "image_type" : 1,
        "sn" : "ef69b450-21bf3bff",
        "type" : "get_img_by_id"
    },
    "cmd" : "dg_json_request",
    "id" : "132156"
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30
body	需要传递给组网模块的数据

body 内的字段说明：

字段名	说明

image_id	图片 id
image_type	图片类型
sn	设备序列号
type	操作组网模块的命令 ● get_img_by_id 得到当前组网内部所有设备配置

结果返回

```
{
    "body" : {
        "image_id" : 79509,
        "image_type" : 1,
        "state" : 404,
        "type" : "get_img_by_id"
    },
    "cmd" : "dg_json_request",
    "id" : "132156",
    "state_code" : 200
}
```

字段名	说明
id	序列字符串（唯一），字符串长度 < 30，等于请求时的 id
body	组网模块处理后返回的数据
state_code	状态码

body 内的字段说明：

字段名	说明
image_id	图片 id
image_type	图片类型
state	状态码
type	操作组网模块的命令 ● get_img_by_id 得到当前组网内部所有设备配置

2.9 用户登录

2.9.1 请求开始登录:start_login

请求命令参数

```
{
  "cmd" : "start_login",
   "id" : "123456"
}
```

结果返回

```
{
        "active_id":0,
        "cmd":"start_login",
        "device_sn":"b77b647f-277f2aa9",
        "ems":[
                {"m_id":0,"name":"none encrypt"},
                {"m_id":1,"name":aes-128-ecb-padding-zeros},
```

```
                {"m_id":2,"name":level 2}
                ],
        "id":"123456",
        "signature":" Rr17TBjPHNYXJR80",
        "state_code":200
}
```

字段名	说明
active_id	当前加密方式的索引
ems	加密方式列表 ● **m_id** 加密方式的索引 ● **name** 加密方式的名字
signature	一个十六位的字符串（用于对传输的密码进行加密，每调用一次 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt/start_login 四个接口中的任意一个接口, signature 都会变 ,重置/修改密码和开启加密需采用最新的 signature ）
state_code:	状态码 ● 200：成功

2.9.2 登录认证: login_authentication

请求命令参数

```
{
    "cmd" : "login_authentication",
    "id" : "13245",
     "authentication" : "12345789"
}
```

结果返回

```
{
    "cmd":"login_authentication",
    "id":"13245",
    "state_code":200
}
```

注：认证失败，直接断开会话连接，需重新建立会话连接

字段名	说明
authentication	用户密码; 使用 HMAC-SHA1 算法对 用户密码进行加密，加密物料使用 signature，再对加密后的数据进行 Base64 生成的字符串。 Signature 可通过 get_ems/reset_encrypt_key/change_encrypt_key/enable_encrypt/start_login 接口获取，需采用最新的 signature
id	序列字符串（唯一），字符串长度 ＜ 30，等于请求时的 id

如

```
    Std::string    user_pass;
    Std::string    signature;
    Std::string    authentication;
    unsigned char digest_buffer[20] = {0};

    HmacSha1((const unsigned char *) user_pass.c_str(), user_pass.size(),
            (const unsigned char *) signature.c_str(),signature.size(), digest_buffer);

    Base64::EncodeFromArray(digest_buffer,    20, &authentication);
```

2.10 语音协议

语音协议主要分为两部分，配置和播放；

注 意：此处的语音为相机设备自带的语音，非外接；

请求命令参数如下格式：

```
{
   "cmd": "playserver_json_request",
   "id": "132156",
   "body": {
         "type": "%s"
         ...
      }
}
```

回复格式为：

```
{
   "cmd": "playserver_json_request",
   "id": "132156",
   “state_code”:200,
   "body": {
         "type": "%s"
         ...
      }
}
```

2.10.1 获取当前语音文件列表：playserver_json_request

请求命令：

```
{
   "cmd" : "playserver_json_request",
   "id":"132156",
   "body":{
      "type":"ps_get_voice_info",
      "voice_type":3
   }
}
```

回复参数：

```
{
    "body" : {
        "type" : "ps_get_voice_info",
        "voice" : [
            {
                "file_name" : "MC53YXY="
            },
            {
                "file_name" : "6bKBLndhdg=="
            },
            {
                "file_name" : "6buRLndhdg=="
            }
        ],
        "voice_type" : 4
    },
    "cmd" : "playserver_json_request",
    "id" : "132156",
    "state_code" : 200
}
```

字段名	说明

type	获取语音文件 **ps_get_voice_info**
file_name	语音文件名称，base64 字符串；
voice_type	文件类型： 用户上传语音目录结构为:/home/admin/video/male, /home/admin/video/female 0. 未知类型 1. 用户语音男声（用户通过 ftp 上传至/home/admin 目录下.wav 文件） 2. 用户语音女声（用户通过 ftp 上传至/home/admin 目录下.wav 文件） 3. 系统语音男声 4. 系统语音女声

2.10.2 设置语音默认参数：playserver_json_request

请求命令：

```
{
   "cmd": "playserver_json_request",
   "id": "132156",
   "body": {
      "type" : "ps_set_voice_config",
      "voice_defalut_interval" : 1,
      "voice_defalut_volume" : 1,
      "voice_defalut_male" : 1
    }
}
```

回复参数：

```
{
    "body" : {
        "type" : "ps_set_voice_config"
    },
    "cmd" : "playserver_json_request",
    "id" : "132156",
    "state_code" : 200
}
```

字段名	说明
voice_defalut_interval	语音文件默认播放间隔，单位为毫秒；
voice_defalut_volume	语音默认音量，范围为[0, 100],现在还没有用，待开放；
voice_defalut_male	默认语音类型：0 男声；1 女声；
type	ps_set_voice_config 设置语音

2.10.3 获取语音默认参数：playserver_json_request

请求命令：

```
{
    "cmd" : "playserver_json_request",
    "id":"132156",
    "body":{
        "type" : "ps_get_voice_config"
     }
}
```

回复结果：

```
{
    "body" : {
        "type" : "ps_get_voice_config",
```

```
            "voice_defalut_interval" : 300,
            "voice_defalut_male" : 1,
            "voice_defalut_volume" : 100
        },
        "cmd" : "playserver_json_request",
        "id" : "132156",
        "state_code" : 200
    }
```

2.10.4 播放语音参数： playserver_json_request

请求命令：

```
    {
        "cmd" : "playserver_json_request",
        "id":"132156",
        "body":{
                "type" : "ps_voice_play",
                "voice" : "agaasg",
                "voice_interval" : 1,
                "voice_volume" : 1,
                "voice_male" : 1
            }
     }
```

回复结果：

```
    {
        "cmd" : "playserver_json_request",
        "id":"132156",
        "state_code":200,
        "body":{
            "type" : " ps_voice_play "
        }
    }
```

字段名	说明
voice	语音信息，utf-8/GBK 编码的 BASE64 编码字符串
voice_interval	语音文件播放间隔
voice_volume	语音文件音量大小
voice_male	语音类型：0 男声；1 女声
type	ps_voice_play 语音播放

2.10.5 请求语音对讲： start_talk

注意：设置语音对讲后，由相机回复出来的消息，是广播到所有的会话连接；
此语音对讲命令只是控制命令，具体的语音对讲信息通过双方协商的端口通信；

请求命令参数

```
{
    "cmd": "start_talk",
    "id" : "1234",
    "body":{
        "window_size":640
    }
}
```

回复结果

```
{
   "body" : {
       "encode_type" : 3,
       "port" : 2707,
       "status" : 1,
       "window_size" : 640
```

```
    },
    "cmd" : "start_talk",
    "id" : "1234",
    "state_code" : 200
}
```

字段名	说明
encode_type	支持的编码类型 1. PCM 2. G711 3. PCM | G711
port	语音对讲通信端口
status	对讲状态 0 空闲 1 忙碌
window_size	服务器窗口大小，每次发送数据包的大小

2.10.6 监听语音对讲：request_talk

注意：此命令，为相机主动向 TCP 连接的会话，主动发起的语音对讲请求，需要 tcp 客户端监听此命令；

监听 cmd 命令参数

```
{
     "cmd": " request _talk",
     "id" : "1234",
     "state_code" : 200,
     “error_msg”:”ok”
}
```

收到此命令时，TCP 客户端作为语音对讲的服务端，也需要回复如下信息：

```
{
    "body" : {
        "encode_type" : 3,
        "port" : 2707,
        "status" : 1,
        "window_size" : 640
    },
    "cmd" : "start_talk",
    "id" : "1234",
    "state_code" : 200
}
```

2.11 屏显协议

注 意：

1.仅当 led_enable 为 1 时，led_content 中的内容才有效；
2. line_content 数组大小与 led_line_num 保持一致；最大不超过 4；
3.仅当 voice_mode 不为 0 时，voice_content 中的内容有效；
4.与网页配置互斥，通过 TCP 配置后，网页配置不再生效，网页配置后，TCP 配置失效；
5.请注意此处的语音为外接 LED 屏显自带的语音；

2.11.1 设置 LED 显示内容：set_led_show

请求命令参数

```
{
     "cmd" : "set_led_show",
     "id" : "12365",
     "body" : {
```

```
        "led_enable" : 1,
        "led_content" :{
          "led_proto" : 1,
          "led_status" : 1,
          "led_refresh_time" : 1,
          "led_line_num" : 4,
          "line_content":
           [{"show_mode" : 1, "show_content" : "5qyi6L+O5YWJ5Li0"},
            {"show_mode" : 1, "show_content" : "5qyi6L+O5YWJ5Li0"},
            {"show_mode" : 1, "show_content" : "5qyi6L+O5YWJ5Li0"},
            {"show_mode" : 1, "show_content" : "5qyi6L+O5YWJ5Li0"}]
        },
        "voice_mode" : 1,
        "voice_content" : {
          "voice_volume":2,
          "voice_welcom": 1,
          "voice_tag" : 1,
          "play_content":"5qyi6L+O5YWJ5Li0"
        },
        "car_info":{
          "park_time" : 32,
          "payment_amount" : 9,
          "car_type" : 1,
          "car_plate" : "5qyi6L+O5YWJ5Li0"
        }
      }
}
```

结果返回

```
{
    "cmd" : "set_led_show",
    "id" : “12365”,
    "state_code":200
}
```

2.11.2 获取 LED 显示内容：get_led_show

请求命令参数

```
{
    "cmd" : " get_led_show",
    "id" : “12365”
}
```

结果返回

```
{
    "cmd" : "get_led_show",
    "id" : “12365”,
    "state_code":200
    "body" : {
      "led_enable" : 1,
      "led_content" :{
        "led_proto" : 1,
        "led_status" : 1,
        "led_refresh_time" : 1,
        "led_line_num" : 4,
        "line_content":
         [{"show_mode" : 1, "show_content" : "5qyi6L+O5YWJ5Li0"},
          {"show_mode" : 1, "show_content" : "5qyi6L+O5YWJ5Li0"},
          {"show_mode" : 1, "show_content" : "5qyi6L+O5YWJ5Li0"},
          {"show_mode" : 1, "show_content" : "5qyi6L+O5YWJ5Li0"}]
      },
```

```
        "voice_mode" : 1,
        "voice_content" : {
           "voice_volume":2,
           "voice_welcom": 1,
           "voice_tag" : 1,
           "play_content":"5qyi6L+O5YWJ5Li0"
        },
        "car_info":{
           "park_time" : 32,
           "payment_amount" : 9,
           "car_type" : 1,
           "car_plate" : "5qyi6L+O5YWJ5Li0"
        }
     }
}
```

2.11.3 获取 LED 数据传输使用串口号：get_led_serial_port

```
{
     "cmd" : " get_led_serial_port",
     "id" : “12365”
}
```

结果返回

```
{
     "cmd" : "get_led_serial_port",
     "use_serial_port" : 1
     "id" : “12365”,
     "state_code":200
}
```

2.11.4 设置 LED 数据传输使用串口号：set_led_serial_port

```
{
     "cmd" : "set_led_serial_port",
     "body" : {
          "use_serial_port" : 1
     },
     "id" : 12365
}
```

结果返回

```
{
     "cmd" : "set_led_serial_port",
     "id" : 12365,
     "state_code":200
}
```

2.11.5 字段信息以及相关枚举

字段名	类型	说明
voice_mode	int	语音播放播放模式：LED_VOICE_PLAY_MODE
voice_content	无	voice_mode 不为 0 时有效
voice_volume	int	音量大小（0 - 9）
voice_welcom	int	欢迎语： LED_VOICE_USER_DEFINE

voice_tag	int	结束语：LED_VOICE_USER_DEFINE
led_enable	int	使能屏显 (0:去使能, 1:使能)
led_content	无	led_enable 为 1 时，内容有效
led_line_num	int	屏幕支持行数(目前最大支持 4 行)
led_proto	int	屏显协议：参考枚举定义： RS485_CTRL_PRO
led_status	int	屏显状态，用户自保存
led_refresh_time	int	屏显内容定时刷新时间[6,15]
line_content	无	屏显每行具体内容,数组大小与 led_line_num 值一致
show_mode	int	屏显显示模式，参考枚举： LED_SCREEN_SHOW_MODE
show_content	string	用户自定义显示（utf-8 的字符串经过 64 位编码之后的内容）
play_content	string	用户自定义播放（utf-8 的字符串经过 64 位编码之后的内容）仅支持 32 省份简称+大写字母+数字+军+警+学+十+百+千+万+年+月+日
use_serial_port	int	使用的串口号（1,2，部分设备支持 3）
state	int	回复状态；200：成功，400：格式错误，500 服务器错误
car_info	无	用户需要显示车辆信息时，有效
park_time	int	停车时间（分钟）
payment_amount	int	收费金额，元
car_type	int	车辆类型：LED_VOICE_CAR_TYPE
car_plate	string	车牌号：UTF-8 经过 64 位编码

相关枚举定义：

1.屏显协议：RS485_CTRL_PRO

```
enum RS485_CTRL_PRO {
    ZS_SERIAL = 0x00,        /* A 臻识科技识别结果输出协议      */
    YB_BX5K1 = 0x01,         /* B 仰邦科技 BX5K1 控制卡协议      */
    XQ_EQ2013 = 0x02,        /* C 谐阔 EQ 系列 2013 控制卡协议    */
    KF_LED = 0X03 ,          /* D 科发 LED 控制卡协议 */
    FK_LED = 0X04 ,          /* E 方控智能 LED 控制卡协议 */
 };
```

2.屏显模式：LED_SCREEN_SHOW_MODE

```
enum LED_SCREEN_SHOW_MODE {
        LED_SHOW_MODE_NONE = 0x00,              // 不显示
        LED_SHOW_MODE_USER_DEFINE = 0x01,    // 显示用户自定义内容
        LED_SHOW_MODE_SYS_TIME = 0x02,        // 显示时间
        LED_SHOW_MODE_FREE_PARKING = 0x04,  // 显示空余车位
        LED_SHOW_MODE_CAR_PLATE = 0x08,                          // 显示车牌号
```

```
    LED_SHOW_MODE_CAR_TYPE = 0x10,                  // 显示车类型
    LED_SHOW_MODE_PARK_TIME = 0x20,                 // 显示停车时间
    LED_SHOW_MODE_CHARGE_MONEY = 0x40,      // 显示收费金额

    LED_SHOW_MODE_MAX,
};
```

3.语音播放模式：LED_VOICE_PLAY_MODE

```
enum LED_VOICE_PLAY_MODE {
    LED_VOIC_PLAY_NONE = 0x00,                  // 不播放语音
    LED_VOICE_PLAY_WELCOM = 0x01,               // 欢迎语
    LED_VOICE_PLAY_CAR_TYPE = 0x02,                 // 车辆类型
    LED_VOICE_PLAY_CAR_PLATE = 0x04,            // 车牌号
    LED_VOICE_PLAY_PARK_TIME = 0x08,            // 停车时间
    LED_VOICE_PLAY_CHARGE_MONEY = 0x10,         // 收费金额
    LED_VOICE_PLAY_TAG = 0x20,                  // 结束语
    LED_VOICE_PLAY_USER_DEFINE = 0x40,          // 自定义语音

    LED_VOICE_PLAY_MAX,
};
```

4. 播放语音的车辆类型：LED_VOICE_CAR_TYPE

```
enum LED_VOICE_CAR_TYPE{
    CAR_TYPE_MONTH_RENT = 1,            //月租车
    CAR_TYPE_TEMP_CAR = 2,                  // 临时车
    CAR_TYPE_NO_PLATE = 3,                  // 无牌车
    CAR_TYPE_BLACKLIST = 4,                 // 黑名单
    CAR_TYPE_MONTH_RENT_EXPIRE = 7,     // 月租车到期
    CAR_TYPE_SPECIAL_CAR = 9,           // 特殊车

    CAR_TYPE_MAX,
};
```

5.播放的欢迎语、结束语

```
enum LED_VOICE_USER_DEFINE {
    USER_DEFINE_VOICE_NONE = 0,                 //
    USER_DEFINE_VOICE_WELCOM_1,                 // 您好
    USER_DEFINE_VOICE_WELCOM_2,             // 欢迎光临
    USER_DEFINE_VOICE_WELCOM_3,             // 欢迎回家

    USER_DEFINE_VOICE_TAG_1,                // 一路顺风
    USER_DEFINE_VOICE_TAG_2,            // 一路平安
    USER_DEFINE_VOICE_TAG_3,                // 欢迎下次光临
```

```
    USER_DEFINE_VOICE_MAX,
};
```