

iSurpass IoT 开放服务接口

V2.16.6

目录

1. 关于 iSurpass 开放服务	14
1.1. iSurpass 开放服务基本架构	14
1.2. iSurpass 开放服务的安装包	14
2. HTTP 基础服务	15
2.1. 认证	15
2.1.1. JWT Token 定义	15
2.1.2. JWT token 使用方式	16
2.2. 网关	16
2.2.1. 获取节点设备列表	16
2.2.2. 获取逻辑节点设备列表	17
2.2.3. (z-wave 网关/无源网关)删除死节点	19
2.2.4. 获取当前连接状态	20
2.2.5. 获取 WIFI 或蜂窝移动网关的信号强度	21
2.2.6. 删除异步请求	22
2.2.7. (无源网关)添加无源设备	23
2.2.8. 执行情景	23
2.2.9. 创建情景	24
2.2.10. 删除情景	25
2.2.11. 清除情景	26
2.2.12. 暂停情景	27
2.2.13. 恢复情景	28
2.2.14. 获取情景列表	29
2.2.15. 获取情景	30
2.2.16. 清除异步请求	31
2.2.17. (Z-wave7 网关)注册 DSK	32
2.2.18. (Z-wave7 网关)注销 DSK	32
2.2.19. (Z-wave7 网关) 获取 RF REGION	33
2.2.20. (Z-wave7 网关) 设置 RF REGION	34

2.3.	锁.....	34
2.3.1.	删除锁用户	35
2.3.2.	添加/修改锁用户	36
2.3.3.	修改锁用户的有效日期时间	37
2.3.4.	修改锁用户的有效星期	38
2.3.5.	获取锁的当前状态.....	39
2.3.6.	远程开锁.....	40
2.3.7.	远程落锁.....	41
2.3.8.	将服务器的系统时间同步到锁.....	42
2.3.9.	获取锁的当前日期时间	43
2.3.10.	获取锁的版本信息.....	44
2.3.11.	清除锁的数据.....	45
2.3.12.	获取锁用户的有效日期时间.....	46
2.3.13.	获取锁用户的有效星期.....	47
2.3.14.	添加卡.....	48
2.3.15.	添加指纹.....	49
2.3.16.	设置时区	50
2.3.17.	停用锁用户.....	51
2.3.18.	恢复锁用户.....	51
2.4.	开关.....	52
2.4.1.	获取开关的当前状态.....	52
2.4.2.	远程开.....	53
2.4.3.	远程关.....	54
2.5.	窗帘	54
2.5.1.	获取窗帘的当前状态.....	55
2.5.2.	远程开.....	55
2.5.3.	远程开(指定百分比).....	56
2.5.4.	远程关.....	57
2.6.	调光灯.....	58
2.6.1.	获取调光灯的当前状态	58
2.6.2.	远程开.....	59
2.6.3.	远程开(指定亮度百分比).....	59
2.6.4.	远程关.....	60
2.7.	门磁/窗磁.....	61

2.7.1.	获取门磁/窗磁的当前状态	61
2.8.	移动感应	62
2.8.1.	获取移动感应的当前状态	62
2.9.	空调	63
2.9.1.	获取空调的当前状态	63
2.9.2.	远程开	65
2.9.3.	远程关	65
2.9.4.	远程设置空调运行模式	66
2.9.5.	远程设置空调温度	67
2.9.6.	远程设置空调风速	68
2.10.	温度计	69
2.10.1.	获取温度计的当前状态	69
2.11.	亮度计	70
2.11.1.	获取亮度计的当前状态	70
2.12.	湿度计	71
2.12.1.	获取湿度计的当前状态	72
2.13.	烟感	73
2.13.1.	获取烟感的当前状态	73
2.14.	漏水感应	74
2.14.1.	获取漏水感应的当前状态	74
2.15.	红外控制	75
2.15.1.	获取红外控制的当前状态	75
2.15.2.	红外学习	76
2.15.3.	远程开	77
2.15.4.	远程关	77
2.15.5.	远程控制	78
2.16.	电表	79
2.16.1.	获取电表的当前状态	79
2.17.	水表	80
2.17.1.	获取水表的当前状态	80
2.18.	人脸识别	81
2.19.	AV 音视频设备	81
2.19.1.	获取音视频设备的当前状态	82
2.19.2.	远程开	82

2.19.3.	远程关.....	83
2.19.4.	远程设置音量.....	83
2.19.5.	远程调低/高音量.....	83
2.19.6.	远程设置静音/取消静音.....	84
2.19.7.	远程控制暂停/恢复播放/停止播放.....	84
2.19.8.	远程控制播放下一首/上一首.....	85
2.19.9.	远程设置播放模式.....	85
2.20.	中控多媒体主机.....	86
2.20.1.	远程获取视频文件列表.....	86
2.20.2.	远程打开视频文件.....	87
2.20.3.	远程获取 PPT 文件列表.....	87
2.20.4.	远程打开 PPT 文件.....	88
2.20.5.	远程热键控制.....	88
2.21.	净水器.....	89
2.21.1.	获取净水器的当前状态.....	89
2.22.	滚筒洗衣机.....	89
2.22.1.	获取洗衣机的当前状态.....	90
2.22.2.	(NB-IoT)远程控制.....	91
2.22.3.	(NB-IoT)远程复位.....	92
2.23.	波轮洗衣机.....	93
2.23.1.	获取洗衣机的当前状态.....	93
2.23.2.	(NB-IoT)远程控制.....	94
2.24.	波轮洗衣机 B.....	94
2.24.1.	获取洗衣机的当前状态.....	95
2.24.2.	(NB-IoT)远程控制.....	96
2.25.	洗鞋机.....	97
2.25.1.	获取洗鞋机的当前状态&参数.....	97
2.25.2.	(NB-IoT)远程控制.....	98
2.25.3.	(NB-IoT)远程设定价格.....	99
2.25.4.	(NB-IoT)远程设定时间参数.....	99
2.25.5.	(NB-IoT)远程设定过程参数.....	100
2.25.6.	(NB-IoT)远程设定特殊功能.....	101
2.26.	干衣机.....	101
2.26.1.	获取干衣机的当前状态.....	101

2.26.2.	(NB-IoT)远程控制	103
2.27.	SOS 紧急求救.....	105
2.27.1.	获取设备的当前状态	105
2.28.	串口设备	106
2.28.1.	获取串口设备的当前状态	106
2.28.2.	远程开	107
2.28.3.	远程关	108
2.28.4.	设置情景	108
2.28.5.	删除情景	109
2.28.6.	运行情景	110
2.28.7.	设置串口通讯超时时间	111
2.28.8.	获取串口通讯超时时间	112
2.28.9.	设置串口通讯波特率	113
2.28.10.	获取串口通讯波特率	114
2.28.11.	远程控制	115
2.29.	路灯	115
2.29.1.	获取路灯的当前状态	116
2.29.2.	远程开	116
2.29.3.	远程开(指定亮度百分比).....	117
2.29.4.	远程关	117
2.29.5.	设定日期定时计划	118
2.29.6.	删除日期定时计划	119
2.29.7.	设定星期定时计划	119
2.29.8.	删除星期定时计划	120
2.30.	电梯	120
2.30.1.	联网授权	121
2.30.2.	删除联网授权	121
2.30.3.	暂停授权	122
2.30.4.	恢复授权	122
2.30.5.	远程呼梯	123
2.30.6.	设置电梯信息	123
2.30.7.	获取电梯信息	124
2.30.8.	获取蓝牙乘梯参数	124
2.30.9.	设置蓝牙乘梯参数	125

2.30.10.	设置蓝牙乘梯灵敏度	125
2.31.	电梯对讲设备	126
2.31.1.	读取配置	126
2.31.2.	下发配置	129
2.31.3.	读取电话本	131
2.31.4.	下发电话本	132
2.31.5.	远程重启	133
2.31.6.	设备升级	133
2.32.	断路器	134
2.32.1.	获取当前状态	134
2.32.2.	远程合闸	135
2.32.3.	远程分闸	136
2.32.4.	设置定时断合	136
2.32.5.	删除定时断合	137
2.32.6.	设置上报周期	137
2.32.7.	设置告警阈值	138
3.	WebSocket 服务	139
3.1.	认证	139
3.2.	请求 z-wave 网关设备进入/退出添加&删除设备状态	139
3.3.	接收设备的 report	140
3.3.1.	请求接收指定 topic 的 report	141
3.3.1.1.	请求接收全部设备 report	141
3.3.1.2.	请求接收指定网关设备的 report	141
3.3.1.3.	请求接收指定网关下的指定节点设备的 report	141
3.3.2.	请求停止接收指定 topic 的 report	141
3.3.2.1.	请求停止接收全部设备 report	141
3.3.2.2.	请求停止接收指定网关下的 report	141
3.3.2.3.	请求停止接收指定网关下的指定节点设备的 report	141
3.4.	设备的 report	142
3.4.1.	网关设备的 report	142
3.4.1.1.	增加节点设备 report	142
3.4.1.2.	删除节点设备 report	143
3.4.1.3.	网关设备上线 report	143
3.4.1.4.	网关设备离线 report	144
3.4.1.5.	(wifi/蜂窝移动)信号强度 report	144
3.4.1.6.	节点故障 report	144
3.4.1.7.	网关设备复位 report	145
3.4.1.8.	网关扩展属性变更 report	145
3.4.1.9.	节点扩展属性变更 report	145

3.4.1.10.	逻辑节点扩展属性变更 report	146
3.4.1.11.	增加无源设备 report	146
3.4.1.12.	(TCP 网关)节点连接 report	146
3.4.1.13.	(TCP 网关)节点断开连接 report	147
3.4.2.	锁的 report	147
3.4.2.1.	锁最新状态 report	147
3.4.2.2.	锁无线信号 report	148
3.4.2.3.	锁电池电量 report	148
3.4.2.4.	开门通知 report	148
3.4.2.5.	关门通知 report	149
3.4.2.6.	锁电池低电量告警 report	149
3.4.2.7.	取消反锁 report	150
3.4.2.8.	锁设备告警 report	150
3.4.2.9.	反锁 report	151
3.4.2.10.	添加/修改锁用户 report	151
3.4.2.11.	修改锁用户有效日期时间 report	152
3.4.2.12.	修改锁用户有效星期时间 report	152
3.4.2.13.	批量修改锁用户有效日期时间 report	153
3.4.2.14.	删除锁用户 report	153
3.4.2.15.	清除锁用户 report	154
3.4.2.16.	设置锁用户名称 report	154
3.4.2.17.	开锁失败 report	154
3.4.2.18.	NB 锁人工唤醒 report	155
3.4.2.19.	获取蓝牙钥匙 report	155
3.4.2.20.	更换蓝牙密钥 report	156
3.4.2.21.	复位蓝牙密钥 report	156
3.4.2.22.	异步批量添加/修改锁用户 report	156
3.4.2.23.	异步批量删除锁用户 report	157
3.4.2.24.	锁的附加门磁的最新状态 report	158
3.4.2.25.	是否允许远程开锁最新状态 report	158
3.4.2.26.	门铃最新状态 report	158
3.4.2.27.	(多个)属性状态 report	159
3.4.2.28.	停用锁用户 report	159
3.4.2.29.	恢复锁用户 report	160
3.4.2.30.	身份证认证结果 report	160
3.4.2.31.	上报密码/卡号 report	161
3.4.2.32.	串口透传 report	161
3.4.3.	开关/无源开关的 report	162
3.4.3.1.	开关最新状态 report	162
3.4.4.	窗帘的 report	162
3.4.4.1.	窗帘最新状态 report	162
3.4.5.	调光灯的 report	163
3.4.5.1.	调光灯最新状态 report	163
3.4.6.	门磁/窗磁的 report	163
3.4.6.1.	门磁/窗磁最新状态 report	163
3.4.6.2.	门磁/窗磁电池电量 report	164
3.4.6.3.	门磁/窗磁电池低电量告警 report	164
3.4.6.4.	门磁/窗磁设备告警 report	164

3.4.6.5.	(多个)属性状态 report.....	165
3.4.7.	移动感应的 report.....	165
3.4.7.1.	进入 report.....	165
3.4.7.2.	离开 report.....	166
3.4.7.3.	移动感应电池电量 report	166
3.4.7.4.	移动感应电池低电量告警 report.....	166
3.4.7.5.	移动感应设备告警 report	167
3.4.8.	空调的 report.....	167
3.4.8.1.	开/关状态 report.....	167
3.4.8.2.	当前室温 report.....	168
3.4.8.3.	当前运行模式 report.....	168
3.4.8.4.	设定温度 report.....	168
3.4.8.5.	设定风速模式 report.....	169
3.4.8.6.	空调电池电量 report.....	169
3.4.8.7.	空调电池低电量告警 report.....	170
3.4.9.	温度计的 report.....	170
3.4.9.1.	温度 report.....	170
3.4.9.2.	温度计电池电量 report.....	171
3.4.9.3.	温度计电池低电量告警 report	171
3.4.10.	亮度计的 report.....	171
3.4.10.1.	亮度 report.....	171
3.4.10.2.	亮度计电池电量 report.....	172
3.4.10.3.	亮度计电池低电量告警 report	172
3.4.11.	湿度计的 report.....	173
3.4.11.1.	湿度 report.....	173
3.4.11.2.	湿度计电池电量 report.....	173
3.4.11.3.	湿度计电池低电量告警 report	174
3.4.12.	烟感的 report	174
3.4.12.1.	烟雾报警 report.....	174
3.4.12.2.	撤销烟雾报警 report.....	174
3.4.12.3.	烟感电池电量 report.....	175
3.4.12.4.	烟感电池低电量告警 report.....	175
3.4.12.5.	设备告警 report.....	176
3.4.12.6.	撤销设备告警 report.....	176
3.4.13.	漏水感应的 report.....	176
3.4.13.1.	漏水报警 report.....	176
3.4.13.2.	撤销漏水报警 report.....	177
3.4.13.3.	漏水感应电池电量 report	177
3.4.13.4.	漏水感应电池低电量告警 report.....	178
3.4.13.5.	设备告警 report.....	178
3.4.13.6.	撤销设备告警 report.....	178
3.4.14.	电表的 report	179
3.4.14.1.	电表读数 report.....	179
3.4.15.	水表的 report	179
3.4.15.1.	水表读数 report.....	179
3.4.16.	(中科奥森)人脸识别设备的 report.....	180

3.4.16.1.	通行 report.....	180
3.4.17.	情景面板的 report.....	180
3.4.17.1.	情景面板触控 report.....	180
3.4.18.	净水器的 report.....	181
3.4.18.1.	滤芯剩余时间 report.....	181
3.4.18.2.	制水状态 report.....	181
3.4.18.3.	制水记录 report.....	182
3.4.19.	SOS 紧急求救设备的 report.....	182
3.4.19.1.	电池电量 report.....	182
3.4.19.2.	电池低电量告警 report.....	183
3.4.19.3.	紧急求救告警 report.....	183
3.4.20.	洗衣机/洗鞋机/干衣机的 report.....	183
3.4.20.1.	刷卡记录 report.....	183
3.4.21.	串口设备的 report.....	184
3.4.21.1.	串口 report.....	184
3.4.22.	蓝牙耳机标的 report.....	184
3.4.22.1.	(多个)属性状态 report.....	184
3.4.23.	燃气感应的 report.....	185
3.4.23.1.	燃气报警 report.....	185
3.4.23.2.	撤销燃气报警 report.....	185
3.4.24.	路灯的 report.....	185
3.4.24.1.	(设备网络状态) 多属性状态 report.....	185
3.4.24.2.	(设备工作状态) 多属性状态 report.....	186
3.4.25.	医药箱的 report.....	187
3.4.25.1.	(设备版本网络) 多属性状态 report.....	187
3.4.25.2.	(设备工作状态) 多属性状态 report.....	187
3.4.25.3.	(设备地理位置) 多属性状态 report.....	188
3.4.25.4.	电池电量 report.....	188
3.4.25.5.	开箱 report.....	188
3.4.25.6.	附加门磁的最新状态 report.....	189
3.4.26.	电梯的 report.....	189
3.4.26.1.	乘梯 report.....	189
3.4.26.2.	无效乘梯 report.....	190
3.4.26.3.	上报卡号 report.....	190
3.4.27.	电梯对讲设备的 report.....	191
3.4.27.1.	(设备上电)多属性状态 report.....	191
3.4.27.2.	(设备 IP 定位)多属性状态 report.....	191
3.4.28.	断路器的 report.....	192
3.4.28.1.	最新状态 report.....	192
3.4.28.2.	功耗读数 report.....	192
3.4.28.3.	电压电流功率 report.....	193
3.4.28.4.	温度 report.....	193
3.4.28.5.	用电安全告警 report.....	193
3.4.28.6.	用电安全撤销告警 report.....	194

3.5.	接收异步请求的操作结果	194
3.6.	请求指纹采集器采集指纹	195
3.7.	OTA 在线升级设备	196
3.8.	远程控制锁	197
4.	HTTP 高级应用服务	198
4.1.	扩展属性服务	198
4.1.1.	设置网关扩展属性	199
4.1.2.	获取网关属性	199
4.1.3.	按扩展属性查询网关	201
4.1.4.	设置节点设备扩展属性	203
4.1.5.	获取节点设备属性	204
4.1.6.	按扩展属性查询节点设备	205
4.1.7.	设置逻辑节点扩展属性	208
4.1.8.	获取逻辑节点属性	208
4.1.9.	按扩展属性查询逻辑节点	210
4.1.10.	添加设备到 app	212
4.1.11.	设置应用属性	213
4.1.12.	获取应用属性	213
4.1.13.	注册应用	214
4.1.14.	更改应用注册信息	214
4.1.15.	注销应用	215
4.1.16.	注册设备	215
4.2.	锁高级服务	216
4.2.1.	获取锁的用户列表	216
4.2.2.	更改指定类型锁用户的有效日期时间	218
4.2.3.	添加/修改密码/临时密码用户	219
4.2.4.	设置锁用户名称	221
4.2.5.	添加卡	222
4.2.6.	添加指纹	223
4.2.7.	添加/修改锁用户	224
4.2.8.	查询锁异步请求历史	226
4.2.9.	添加/修改蓝牙/二维码用户	229
4.2.10.	修改蓝牙/二维码用户的有效日期时间	230
4.2.11.	修改蓝牙/二维码用户的有效星期	231

4.2.12.	删除蓝牙/二维码用户.....	232
4.2.13.	获取蓝牙/二维码用户钥匙	233
4.2.14.	更换蓝牙密钥	234
4.2.15.	复位蓝牙密钥	236
4.2.16.	按 Owner 获取锁用户列表	237
4.2.17.	异步批量添加/修改锁用户	239
4.2.18.	异步批量删除锁用户	240
4.2.19.	蓝牙加卡	241
4.2.20.	获取酒店锁开锁密码	242
4.2.21.	蓝牙加密码用户	243
4.2.22.	蓝牙将服务器的系统时间同步到锁.....	244
4.2.23.	添加蓝牙钥匙扣用户	244
4.2.24.	蓝牙加卡用户	245
4.2.25.	蓝牙设置锁用户的有效星期时间	246
4.2.26.	蓝牙设置锁用户的有效日期时间	246
4.2.27.	蓝牙清除锁用户的有效日期时间	247
4.2.28.	蓝牙删除锁用户	248
4.2.29.	获取待下发的锁用户异步请求.....	248
4.2.30.	蓝牙获取锁时间	249
4.2.31.	蓝牙获取锁用户列表	249
4.2.32.	蓝牙获取锁用户类型	250
4.2.33.	蓝牙让锁进入 Z-Wave 添加/删除设备状态	251
4.2.34.	蓝牙清除锁数据	251
4.3.	历史记录查询服务.....	252
4.3.1.	(Redis5)查询设备 report.....	252
4.3.2.	(InfluxDB)查询设备 report.....	255
4.3.3.	查询异步请求.....	258
4.3.4.	(InfluxDB)查询远程请求日志 report	259
4.4.	TCP 网关节点设备管理服务.....	259
4.4.1.	注册节点设备	259
4.4.2.	注销节点设备	261
4.5.	滚筒洗衣机高级服务	261
4.5.1.	蓝牙控制洗衣机	262
4.5.2.	蓝牙复位洗衣机	263

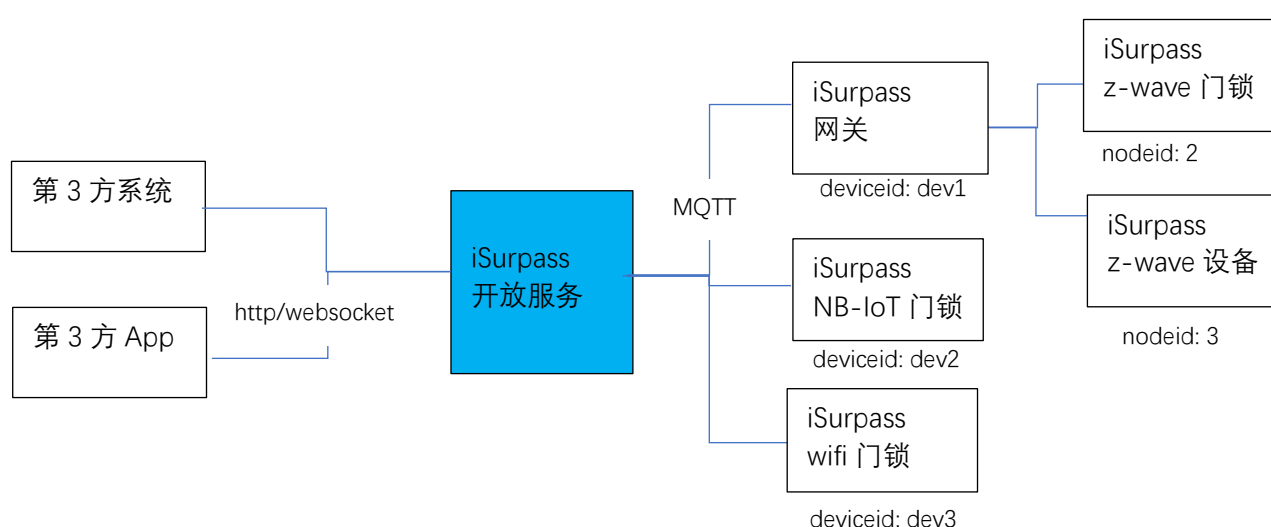
4.5.3.	蓝牙获取洗衣机当前状态	264
4.6.	波轮洗衣机高级服务	265
4.6.1.	蓝牙控制洗衣机	265
4.6.2.	蓝牙获取洗衣机当前状态	266
4.7.	波轮洗衣机 B 高级服务	266
4.7.1.	蓝牙控制洗衣机	266
4.7.2.	蓝牙获取洗衣机当前状态	267
4.8.	洗鞋机高级服务	268
4.8.1.	蓝牙控制洗鞋机	268
4.8.2.	蓝牙设定价格	269
4.8.3.	蓝牙设定时间参数	269
4.8.4.	蓝牙设定过程参数	270
4.8.5.	蓝牙设定特殊功能	271
4.8.6.	蓝牙发送键值	271
4.8.7.	蓝牙获取洗鞋机当前状态	272
4.8.8.	蓝牙获取洗鞋机价格	272
4.8.9.	蓝牙获取时间参数	273
4.8.10.	蓝牙获取过程参数	273
4.8.11.	蓝牙获取特殊功能设定	274
4.9.	干衣机高级服务	275
4.9.1.	蓝牙控制干衣机	275
4.9.2.	蓝牙获取干衣机当前状态	277
4.10.	其他高级服务	277
4.10.1.	强制删除设备	277
4.11.	文件服务	278
4.11.1.	上传文件	278
4.12.	医药箱高级服务	279
4.12.1.	获取蓝牙开锁密码	279
4.12.2.	蓝牙控制设备指令	280
4.13.	批处理	280
4.13.1.	执行批处理	280
4.13.2.	中止批处理	281
5.	(4G CAT1)人脸设备	283
5.1.	http 基础服务	283

5.1.1.	添加人员	283
5.1.2.	删除人员	284
5.1.3.	清空人员	284
5.1.4.	获取人员列表	285
5.1.5.	获取人员名称	285
5.1.6.	获取人员相片	286
5.1.7	下发卡	287
5.1.8	绑定蓝牙门锁	287
5.1.9	蓝牙透传数据	288
5.1.10	下发取电开关	289
5.1.11	冻结/解冻用户	289
5.1.12	更改休眠壁纸	290
5.1.13	修改人脸配置	290
5.2.	http 高级服务	291
5.2.1.	设置 owner 和 name	291
5.2.2.	设置 name	292
5.2.3.	获取人脸设备列表	292
5.2.4.	清除 owner	293
5.3.	人脸设备 report	294
5.3.1.	上线 report	294
5.3.2.	离线 report	294
5.3.3.	通行 report	294
5.4.	人脸设备 mongodb 存储	295
5.4.1.	人脸设备 facedevice	295
5.4.2.	人脸特征 facefeatures	295
A.	设备类型及其转换协议一览表	296

1. 关于 iSurpass 开放服务

iSurpass 开放服务将智能门锁的服务封装为简单易用的 http/websocket 协议、json 数据格式，第 3 方的系统或 App 很容易与它集成。

1.1. iSurpass 开放服务基本架构



- 1) iSurpass 开放服务对第 3 方系统/App 应用提供 http/websocket 接口：http 接口用来接收第 3 方系统/App 的远程请求，websocket 用于远程控制网关设备进入/退出添加&删除设备状态、推送设备的 report 给第 3 方系统/App；
- 2) iSurpass 开放服务与 iSurpass 网关/设备之间采用 MQTT 协议通讯；
- 3) iSurpass 网关(或连接 iSurpass 开放服务的 NB-IoT 智能门锁、wifi 门锁)上需配置 iSurpass 开放服务的服务器地址；
- 4) iSurpass 开放服务可单机部署，也可多服务器部署。

1.2. iSurpass 开放服务的安装包

iSurpass 开放服务的安装包可提供给第 3 方安装和集成，有以下版本：

-  for Windows: 32bit or 64bit
-  for Linux: 32bit or 64bit
-  for 树莓派

2. HTTP 基础服务

2.1. 认证

采用 JWT 认证, see:

<https://baijiahao.baidu.com/s?id=1608021814182894637>

<https://jwt.io/>

2.1.1. JWT Token 定义

■ JWT header

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

■ JWT payload

```
{  
  "appkey": "<AppKey>",          #第 3 方应用的 AppKey  
  "timestamp": "<TimeStamp>"    #当前时间戳  
}
```

■ JWT signature

HMACSHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload), <AppSecret>)
AppSecret 为第 3 方应用的密钥

■ JWT token

base64UrlEncode(header)+"."+base64UrlEncode(payload)+"."+signature

示例:

JWT header:

```
{"alg":"HS256","typ":"JWT"}
```

JWT payload:

```
{"appkey":"iSurpassApp","timestamp":"1234567890"}
```

AppSecret:

iSurpass2018

JWT token:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBBrZXkiOiJpU3VycGFzc0FwcClzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0

2.1.2. JWT token 使用方式

所有 HTTP 请求 Header 带 Authorization 字段, 值为:
Bearer <JWT token>

2.2. 网关

url: http(s)://<ip>:<port>/gateway

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.2.1. 获取节点设备列表

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetNodeList",
  "deviceid": "<deviceid>" //网关设备 ID
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功, 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "NodeList": [
      {
        "NodeID": "<nodeid>", //节点 id
        "NodeType": "<devicetype>", //节点设备类型, 如 lock
        "ProtocolType": "<protocoltype>", //协议类型, 如: lock、nblock、wifilock
        "DeviceModel": "<DeviceModel>" //设备厂商型号 ID
      }
      ...
    ]
  }
}
```

【注】

1) 该接口适用于网关(z-wave 网关、TCP 网关), 不适用于单一设备(WIFI 锁、NB 锁、蓝牙锁、指纹采集器、NB 电表、NB 水表等)

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.GetNodeList",
    "deviceid": "iSurpass2018100000"
  }'
//返回结果:
{
  "data": {
    "NodeList": [
      {
        "NodeID": 2,
        "NodeType": "lock",
        "ProtocolType": "lock"
      },
      {
        "NodeID": 3,
        "NodeType": "lock",
        "ProtocolType": "lock"
      },
      {
        "NodeID": 21,
        "NodeType": "switch",
        "ProtocolType": "switch"
      },
      {
        "NodeID": 22,
        "NodeType": "switch",
        "ProtocolType": "switch"
      }
    ]
  },
  "resultcode": 1
}

```

2.2.2. 获取逻辑节点设备列表

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.GetLogicNodeList",
  "deviceid": "<deviceid>"           //网关设备 ID
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,           //1-成功, 非 1-失败
}

```

```

"errorinfo": "",          //错误信息
"errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
"data": {
    "LogicNodeList": [
        {
            "LogicNodeID": "<logicnodeid>", //逻辑节点 id
            "LogicNodeType": "<devicetype>", //逻辑节点设备类型, 如 lock
            "ProtocolType": "<protocoltype>" //协议类型, 如: lock、nblock、wifilock
        }
        ...
    ]
}
}

```

【注】

1) 该接口适用于网关(z-wave 网关、TCP 网关), 不适用于单一设备(WIFI 锁、NB 锁、蓝牙锁、指纹采集器、NB 电表、NB 水表等)

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d {
    "method": "thing.service.GetLogicNodeList",
    "deviceid": "iSurpass2018100000"
  }

```

//返回结果:

```

{
  "resultcode": 1,
  "data": {
    "LogicNodeList": [
      {
        "LogicNodeID": "iRemote7005000000020-4-0-",
        "LogicNodeType": "lock",
        "ProtocolType": "lock"
      },
      {
        "LogicNodeID": "iRemote7005000000020-22-0-",
        "LogicNodeType": "switch",
        "ProtocolType": "switch"
      },
      {
        "LogicNodeID": "iRemote7005000000020-21-1-",
        "LogicNodeType": "switch",
        "ProtocolType": "switch"
      },
      {
        "LogicNodeID": "iRemote7005000000020-21-2-",
        "LogicNodeType": "switch",
        "ProtocolType": "switch"
      }
    ]
  }
}

```

2.2.3. (z-wave 网关/无源网关)删除死节点

【注】此接口仅适应于删除网关上的死节点(节点对应的设备已长时间离网)，删除活在节点的方式的参见: [请求网关设备进入/退出添加&删除设备状态](#)

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.RemoveDeadNode",
  "deviceid": "<deviceid>", //网关设备 ID
  "params": {
    "NodeID": "<NodeID>" //死节点 ID
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-删除成功, 9-该设备不是死设备 10-删除失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 该接口仅适用于 z-wave 网关、无源网关

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcGlzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.RemoveDeadNode",
    "deviceid": "iSurpass2018100000",
    "params": {
      "NodeID": 4
    }
  }'
```

//返回结果:

```
{
  "resultcode": 1
}
```

2.2.4. 获取当前连接状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetConnectionStatus",
  "deviceid": "<deviceid>", //网关设备 ID
  "params": {
    "ForceRefresh": "<true/false> //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "Online": "<true/false>", //最新状态: true-在线 false-离线
    "LastestDateTime": "<yyyy-mm-dd HH:MM:SS>", //最新状态日期时间
    "Lastest_ts": "<timestamp>", //最新状态 Unix 时间戳(毫秒)
    "IP": "<ip>", //设备的互联网 IP
    "Network": "<network type>", //网络类型: LAN、WIFI、2G
    "Version": "<software version>", //设备的软件版本号
    "SSID": "<wifi ssid>", //WIFI SSID
    "WIFI_IP": "<wifi ip>", //WIFI IP
    "WIFI_MAC": "<wifi mac>", //WIFI MAC
    "RSSI": "<value>" //wifi 或蜂窝移动信号强度值
  }
}
```

【注】

1) 该接口仅适用于 z-wave 网关

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBZXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.GetConnectionStatus",
    "deviceid": "iRemote7005000000020"
  }'
//返回结果:
{
  "resultcode": 1,
  "data": {
    "IP": "119.123.42.149",
    "LastestDateTime": "2019-07-09 11:13:57",
```

```

        "Lastest_ts": 1562642037,
        "Network": "WIFI",
        "Online": true,
        "RSSI": 98,
        "SSID": "Tplink_Office_2.4G",
        "Version": "v1.0.20",
        "WIFI_IP": "192.168.5.6",
        "WIFI_MAC": "F0FE6B50B7E3"
    }
}

```

2.2.5. 获取 WIFI 或蜂窝移动网关的信号强度

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.GetRSSI",
    "deviceid": "<deviceid>"           //网关设备 ID
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,           //1-成功 非 1-失败
    "errorinfo": "",           //错误信息
    "errorinfo_cn": "",        //本地化错误信息(Accept-Language 相关)
    "deviceid": "<deviceid>",
    "report": "thing.event.property.post",
    "type": "notification",
    "params": {
        "property": "RSSI",
        "value": <value>       //信号强度值
    }
}

```

【注】

1) 该接口仅适用于 z-wave 网关

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Accept: */*' \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCl6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.GetRSSI",
    "deviceid": "iSurpass2019100002"
  }'

```

//返回结果:

```

{
    "RawData": "64070005006D000121",

```

```

    "deviceid": "iSurpass2019100002",
    "params": {
      "property": "RSSI",
      "value": 80
    },
    "report": "thing.event.property.post",
    "resultcode": 1,
    "type": "notification"
  }
}

```

2.2.6. 删除异步请求

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.RemoveAsyncRequest",
  "deviceid": "<deviceid>",           //网关设备 ID
  "params": {
    "Receipt": "<receipt>"           //异步调用的回执 id
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,           //1-删除成功
  "errorinfo": "",           //错误信息
  "errorinfo_cn": ""        //本地化错误信息(Accept-Language 相关)
}

```

[示例]

```

curl -X GET \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.RemoveAsyncRequest",
    "deviceid": "iSurpass2019100041",
    "params": {
      "Receipt": "lock-1560226438542"
    }
  }'
//返回结果:
{
  "resultcode": 1
}

```

【注】

1) 该接口并不能回撤正在下发队列中的请求，最好是在 NB-IoT 等电池供电设备处于休眠状态时调用该接口

2) 该接口暂不支持同一个设备的并发请求，同一设备的两次删除异步请求调用需要相隔 1 秒以上;

2.2.7. (无源网关)添加无源设备

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.AddNaturalDevice",
  "deviceid": "<deviceid>", //网关设备 ID
  "params": {
    "DeviceCode": "<code>", //无源设备特性码, hex 串
    "DeviceKeyCount": <keycount> //无源设备有几个键
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-删除成功
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 该接口仅适用于无源网关

2.2.8. 执行情景

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.RunScene",
  "deviceid": "<deviceid>", //网关设备 ID
  "params": {
    "SceneID": <scene id> //情景 id
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-删除成功
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 该接口仅适用于 z-wave 网关

[示例]

```
curl -X GET \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBBN-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.RunScene",
    "deviceid": "iRemote7005000000020",
    "params": {
      "SceneID": 1
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.2.9. 创建情景

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.NewScene",
  "deviceid": "<deviceid>", //网关设备 ID
  "params": {
    "SceneName": "<scene name>", //情景名称
    "CmdList": [ //情景指令列表(最多 20 条)
      {
        "LogicNodeType": "<logicnode type>", //逻辑设备类型
        "Cmd": "<json>", //执行指令
        "Delay": "<seconds>" //执行时延(秒)
      },
      ...
    ],
    "Trigger": { //情景触发设备, 可选项
      "logicnodeid": "<logicnodeid>", //逻辑节点 ID
      "report": "<report>", //事件 Report Name
      "params": {"<key>": "<val>"} //事件参数值
    }
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-删除成功
  "errorinfo": "", //错误信息
}
```

```

        "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
        "params": {
            "SceneID": <sceneid> //情景 ID
        }
    }
}

```

【注】

1) 该接口仅适用于 z-wave 网关

[示例]

```

curl -X GET \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.NewScene",
    "deviceid": "iRemote7005000000020",
    "params": {
      "CmdList": [
        {
          "LogicNodeType": "switch",
          "Cmd": {
            "method": "thing.service.Turnon",
            "logicnodeid": "iRemote7005000000020-18-1-"
          },
          "Delay": 0
        },
        {
          "LogicNodeType": "switch",
          "Cmd": {
            "method": "thing.service.Turnon",
            "logicnodeid": "iRemote7005000000020-18-2-"
          },
          "Delay": 0
        }
      ]
    }
  }'
//返回结果:
{
  "resultcode": 1,
  "params": {
    "SceneID": 1
  }
}

```

2.2.10. 删除情景

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```
{
  "method": "thing.service.DeleteScene",
  "deviceid": "<deviceid>",          //网关设备 ID
  "params": {
    "SceneID": <scene id>           //情景 id
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-删除成功
  "errorinfo": "",         //错误信息
  "errorinfo_cn": ""       //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 该接口仅适用于 z-wave 网关

[示例]

```
curl -X GET \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.DeleteScene",
    "deviceid": "iSurpass2019100041",
    "params": {
      "SceneID": 1
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.2.11. 清除情景

方法: post

请求头(header):

```
Accept-Language: <language>          //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.ClearScene",
  "deviceid": "<deviceid>"           //网关设备 ID
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-删除成功
  "errorinfo": "",         //错误信息
}
```

```
    "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 该接口仅适用于 z-wave 网关

[示例]

```
curl -X GET \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCl6IjEyMzQ1Njc4OTAifQ.KRLkBBN-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.ClearScene",
    "deviceid": "iSurpass2019100041",
    "params": {
      "SceneID": 1
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.2.12. 暂停情景

方法: post

请求头(header):

```
Accept-Language: <language>    //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.PauseScene",
  "deviceid": "<deviceid>",    //网关设备 ID
  "params": {
    "SceneID": <scene id>    //情景 id
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,    //1-删除成功
  "errorinfo": "",    //错误信息
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 该接口仅适用于 z-wave 网关

[示例]

```
curl -X GET \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Authorization: Bearer
```

```

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAif
Q.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
-H 'cache-control: no-cache' \
-d '{
  "method": "thing.service.PauseScene",
  "deviceid": "iSurpass2019100041",
  "params": {
    "SceneID": 1
  }
}'
//返回结果:
{
  "resultcode": 1
}

```

2.2.13. 恢复情景

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.RestoreScene",
  "deviceid": "<deviceid>",           //网关设备 ID
  "params": {
    "SceneID": <scene id>           //情景 id
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,           //1-删除成功
  "errorinfo": "",           //错误信息
  "errorinfo_cn": ""         //本地化错误信息(Accept-Language 相关)
}

```

【注】

1) 该接口仅适用于 z-wave 网关

[示例]

```

curl -X GET \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAif
Q.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.RestoreScene",
    "deviceid": "iSurpass2019100041",
    "params": {
      "SceneID": 1
    }
  }'

```

```
//返回结果:
{
  "resultcode": 1
}
```

2.2.14. 获取情景列表

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.GetSceneList",
  "deviceid": "<deviceid>" //网关设备 ID
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,
  "data": {
    "SceneIDList": [
      {
        "SceneID": <scene id>
      },
      ...
    ]
  }
}
```

【注】

1) 该接口仅适用于 z-wave 网关

[示例]

```
curl -X GET \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.GetSceneList",
    "deviceid": "iRemote2005000014219"
  }'
```

//返回结果:

```
{
  "data": {
    "SceneIDList": [
      {
        "SceneID": 8
      },
      {
        "SceneID": 9
      }
    ]
  }
}
```

```

    },
    "resultcode": 1
}

```

2.2.15. 获取情景

方法：post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.GetScene",
  "deviceid": "<deviceid>",           //网关设备 ID
  "params": {
    "SceneID": "<scene id>"           //情景 id
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,
  "report": "thing.event.GetScene.post",
  "deviceid": "<deviceid>",
  "params": {
    "SceneName": "<scene name>",       //情景名称
    "Cmdlist": [
      {
        "LogicNodeType": "<LogicNodeType>",
        "Cmd": "<json>",
        "Delay": "<delay time>"       //单位：秒
      }
    ],
    "Trigger": {                       //情景触发设备，可选项
      "logicnodeid": "<logicnodeid>",   //逻辑节点 ID
      "report": "<report>",             //事件 Report Name
      "params": { "<key>": "<val>" }     //事件参数值
    },
    "Status": "<active/pause>",        //情景状态：active-正常状态 pause-暂停状态
  }
}

```

【注】

1) 该接口仅适用于 z-wave 网关

[示例]

```

curl -X GET \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.GetScene",
    "deviceid": "iRemote7005000000020",

```

```

    "params": {
      "SceneID": 3
    }
  }
}
//返回结果:
{
  "deviceid": "iRemote7005000000020",
  "params": {
    "CmdList": [
      {
        "Cmd": {
          "logicnodeid": "iRemote7005000000020-19-1-",
          "method": "thing.service.Turnoff"
        },
        "Delay": 0,
        "LogicNodeType": "switch"
      },
      {
        "Cmd": {
          "logicnodeid": "iRemote7005000000020-19-2-",
          "method": "thing.service.Turnoff"
        },
        "Delay": 0,
        "LogicNodeType": "switch"
      }
    ],
    "SceneName": "中文"
  },
  "report": "thing.service.GetScene",
  "resultcode": 1
}

```

2.2.16. 清除异步请求

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.ClearAsyncRequest",
  "deviceid": "<deviceid>"             //网关设备 ID
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,           //1-删除成功
  "errorinfo": "",          //错误信息
  "errorinfo_cn": ""        //本地化错误信息(Accept-Language 相关)
}

```

[示例]

```
curl -X GET \
  http(s)://112.74.20.113:8888/gateway \
  -H 'Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.ClearAsyncRequest",
    "deviceid": "iSurpass2019100041"
  }'
//返回结果:
{
  "resultcode": 1
}
```

【注】

- 1) 该接口并不能回撤正在下发队列中的请求，最好是在 NB-IoT 等电池供电设备处于休眠状态时调用该接口；
- 2) 该接口将清除指定设备当前时间之前创建的所有待下发的异步请求

2.2.17. (Z-wave7 网关)注册 DSK

方法：post

请求头(header)：

Accept-Language: <language> //en、zh-CN
 Authorization: Bearer <JWT token>

请求参数：

```
{
  "method": "thing.service.RegistDSK",
  "deviceid": "<deviceid>", //网关设备 ID
  "params": {
    "DSK": "<device dsk>" //long range 设备的 dsk
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-删除成功
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 该接口仅适用于 Z-wave7 网关(_supportdsk 属性值为 true)

2.2.18. (Z-wave7 网关)注销 DSK

方法：post

请求头(header)：

Accept-Language: <language> //en、zh-CN
 Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.UnRegistDSK",
  "deviceid": "<deviceid>",          //网关设备 ID
  "params": {
    "DSK": "<device dsk>"           //long range 设备的 dsk
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-删除成功
  "errorinfo": "",          //错误信息
  "errorinfo_cn": ""        //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 该接口仅适用于 Z-wave7 网关(_supportdsk 属性会上为 true)
- 2) 注销 DSK 成功后, 会自动删除相应的节点设备

2.2.19. (Z-wave7 网关) 获取 RF REGION

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetRFRegion",
  "deviceid": "<deviceid>"           //网关设备 ID
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
  "deviceid": "<deviceid>",
  "report": "thing.event.property.post",
  "type": "notification",
  "params": {
    "property": "RFRegion",
    "value": <number>
  }
}
```

RF REGION value:

0 EU
1 US
2 ANZ
3 HK
5 IL
7 RU
8 CN

9 US_LR

32 JP

33 KR

【注】

1) 该接口仅适用于 Z-wave7 网关(_supportdsk 属性值为 true)

2.2.20. (Z-wave7 网关) 设置 RF REGION

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetRFRegion",
  "deviceid": "<deviceid>", //网关设备 ID
  "params": {
    "RFRegion": <number>
  }
}
```

RF REGION value:

0 EU

1 US

2 ANZ

3 HK

5 IL

7 RU

8 CN

9 US_LR

32 JP

33 KR

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-删除成功
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 该接口仅适用于 Z-wave7 网关(_supportdsk 属性值为 true)

2.3. 锁

http(s)://<ip>:<port>/lock

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.3.1. 删除锁用户

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.DeleteUser",
  "async": false, //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
  "params": {
    "UserType": <UserType>, //1-密码 2-指纹 3-MF 卡 4-身份证 7-人脸
    "UserID": <UserID> //锁用户 ID
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.DeleteUser",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params": {
      "UserType": 1,
      "UserID": 10
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.3.2. 添加/修改锁用户

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.AddUser",
  "async": false, //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
  "params": {
    "UserType": "<UserType>", //1-密码 2-指纹 3-MF 卡 4-身份证 7-人脸
    "UserLevel": "<UserLevel>", //1-管理员 2-普通用户 4-一次性临时密码(UserType 必须为 1)
    "UserID": "<UserID>", //UserID 规则:
      管理员: 1~9
      普通用户: 10~230
      一次性临时密码: 243~252
    "UserPassword": "<UserPassword>", //密码串/指纹数据串/卡数据/二维码内容文本
      卡数据: MF 卡, 4byte 值转换为十六进制串
      身份证卡, 8byte 值转换为十六进制串
    "FaceFileURL": "<URL path>", //人脸照片文件的 URL path
    "ValidWeekDays": "<SumWeekdayValue>", //有效星期累加值: 如 5 表示星期天和星期二有效
      1-星期天 2-星期一 4-星期二 8-星期三
      16-星期四 32-星期五 64-星期六
    "ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间, 多个时间段,分隔
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
    "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
    "CustomInfo": "<HEX>" //定制数据(HEX 字符串)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 0-用户已存在 2-锁用户数超上限
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1
- 3) 目前仅 protocoltype 为 entranceb 的设备支持 CustomInfo: 64 个 HEX 字符
- 4) 仅部分设备 ValidWeekTime 支持多个时间段

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
```

```

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
-d '{
  "method": "thing.service.AddUser",
  "deviceid": "iSurpass2018100000",
  "nodeid": 2,
  "params": {
    "UserType": 1,
    "UserLevel": 2,
    "UserID": 10,
    "UserPassword": "5678",
    "ValidBeginDateTime": "2018-10-27 14:06",
    "ValidEndDateTime": "2018-12-27 23:59",
    "ValidWeekDays": 127,
    "ValidWeekTime": "00:00~23:59"
  }
}
//返回结果:
{
  "resultcode": 1
}

```

2.3.3. 修改锁用户的有效日期时间

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.UpdateUserValidDateTime",
  "async": false,                        //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid": <nodeid>,                    //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID>,                  //锁用户 ID
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
    "ValidEndDateTime": "<YYYY-MM-DD HH:MM>"   //有效期的终止日期时间
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,      //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "",      //错误信息
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.UpdateUserValidDateTime",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params": {
      "UserID": 10,
      "ValidBeginDateTime": "2018-10-27 14:06",
      "ValidEndDateTime": "2018-12-27 23:59"
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.3.4. 修改锁用户的有效星期

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.UpdateUserValidWeekTime",
  "async": false, //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID>, //锁用户 ID
    "ValidWeekDays": <SumWeekdayValue>, //有效星期累加值: 如 5 表示星期天和星期二有效
    //1-星期天 2-星期一 4-星期二 8-星期三
    //16-星期四 32-星期五 64-星期六
    "ValidWeekTime": "<HH:MM~HH:MM>" //有效星期的起止时间
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)

2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true，nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcCI6ImlnRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.UpdateUserValidWeekTime",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params":{
      "UserID": 10,
      "ValidWeekDays": 96,
      "ValidWeekTime": "00:00~23:59"
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.3.5. 获取锁的当前状态

方法: post

ForceRefresh
Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
  "params": {
    "ForceRefresh": <true/false> //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": <true/false>, //锁当前是否是故障状态: true-故障状态 false-正常状态
    "Locked": <true/false>, //锁当前是否是落锁状态: true-落锁状态 false-开锁状态
    "LockedFromInside": <true/false>, //锁当前是否反锁: true-反锁状态 false-未反锁状态
    "BatteryPercentage": <percent>, //电池电量百分比: 0~100
    "CardNumber": "<card number>" //((NB 锁)NB 卡卡号
  }
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.property.get",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2
  }'
//返回结果:
{
  "resultcode": 1,
  "data": {
    "BatteryPercentage": 85,
    "Locked": true
  }
}
```

【注】仅 Z-Wave 锁支持 ForceRefresh 设为 true

2.3.6. 远程开锁

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.OpenLock",
  "async": false,                        //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid": <nodeid>,                    //智能锁在网关设备上的节点 id
  "params": {
    "Mode": <mode>,                      //开锁模式: 0-常开 1-自动回锁
    "AutoCancel": <seconds>             //n 秒后自动取消请求, 仅 async 为 true 时有效, 缺省为 0(不自动取消)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                      //1-成功 非 1-失败
  "receipt": "<receipt>",                //异步调用的回执 id
  "errorinfo": "",                       //错误信息
  "errorinfo_cn": ""                     //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.OpenLock",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params": {
      "Mode": 1
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.3.7. 远程落锁

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.CloseLock",
  "async": false,                //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>",    //网关设备 ID
  "nodeid": <nodeid>           //智能锁在网关设备上的节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,              //1-成功 非 1-失败
  "receipt": "<receipt>",      //异步调用的回执 id
  "errorinfo": "",              //错误信息
  "errorinfo_cn": ""           //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
```

```

-d '{
  "method": "thing.service.CloseLock",
  "deviceid": "iSurpass2018100000",
  "nodeid": 2
}'
//返回结果:
{
  "resultcode": 1
}

```

2.3.8. 将服务器的系统时间同步到锁

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.SetNodeSystemTime",
  "deviceid": "<deviceid>",             //网关设备 ID
  "nodeid": <nodeid>                   //智能锁在网关设备上的节点 id
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,           //1-成功 非 1-失败
}

```

[示例]

```

curl -X POST \
  http(s)://dev.isurpass.com.cn:8888/lock \
  -H 'Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.SetNodeSystemTime",
    "deviceid": "iRemote2005000007242",
    "nodeid": 6
  }
'
//返回结果:
{
  "RawData": "1E0800050001000100",
  "resultcode": 1
}

```

2.3.9. 获取锁的当前日期时间

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetNodeSystemTime",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>" //智能锁在网关设备上的节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "report": "thing.event.NodeSystemTime.post",
  "type": "notification",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑设备 id
  "params": {
    "Datetime": "<Y-M-D h:m:s>", //锁的当前日期时间
    "UTCTimestamp": <timestamp>, //UTC 时间戳(秒)
    "Timezone": <timezone> //锁的时区
  }
}
```

[示例]

```
curl -X POST \
  http(s)://dev.isurpass.com.cn:8888/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcGlzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.GetNodeSystemTime",
    "deviceid": "iRemote2005000007242",
    "nodeid": 6
  }
'
```

//返回结果:

```
{
  "deviceid": "iRemote2005000007242",
  "id": 0,
  "logicnodeid": "iRemote2005000007242-6-0-",
  "nodeid": 6,
  "params": {
    "Datetime": "2019-3-1 17:47:9"
  },
  "report": "thing.event.NodeSystemTime.post",
  "type": "notification"
}
```

2.3.10. 获取锁的版本信息

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.GetNodeVersion",
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid":  <nodeid>                    //智能锁在网关设备上的节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "report": "thing.event.NodeVersion.post",
  "type": "notification",
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid":  <nodeid>,                  //智能锁在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>",        //逻辑设备 id
  "params": {
    "Version": "<versioninfo>"          //锁的版本信息
  }
}
```

[示例]

```
curl -X POST \
  http(s)://dev.isurpass.com.cn:8888/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBBN-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
"method": "thing.service.GetNodeVersion",
"deviceid": "iRemote70050000000020",
"nodeid":105
}'
```

//返回结果:

```
{
  "deviceid": "iRemote70050000000020",
  "id": 0,
  "logicnodeid": "iRemote70050000000020-105-0-",
  "nodeid": 105,
  "params": {
    "Version": "000E0102333132000000000000000000"
  },
  "report": "thing.event.NodeVersion.post",
  "type": "notification"
}
```

2.3.11. 清除锁的数据

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.ClearNodeData",
  "async": false, //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
  "params": {
    "Mode": <Mode> // 0-恢复出厂 1-清空密码 2-清空卡片 3-清空指纹 9-蓝牙/二维码 10-清空人脸
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1
- 3) 当 Mode 参数值为 9 时不会下发指令到锁, 非 9 时会下发指令到锁

[示例]

```
curl -X POST \
  http(s)://dev.isurpass.com.cn:8888/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.ClearNodeData",
    "deviceid": "iRemote7005000000020",
    "nodeid": 105,
    "params": {
      "Mode": 1
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.3.12. 获取锁用户的有效日期时间

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetUserValidDateTime",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
  "params": {
    "UserID": "<UserID>" //锁用户 ID
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "params": {
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
    "ValidEndDateTime": "<YYYY-MM-DD HH:MM>" //有效期的终止日期时间
  }
}
```

[示例]

```
curl -X GET \
  http(s)://dev.isurpass.com.cn:8888/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.GetUserValidDateTime",
    "deviceid": "iRemote7005000000020",
    "nodeid": 123,
    "params": {
      "UserID": 11
    }
  }'
```

//返回结果:

```
{
  "deviceid": "iRemote7005000000020",
  "id": 0,
  "logicnodeid": "iRemote7005000000020-123-0-",
  "nodeid": 123,
  "params": {
    "ValidBeginDateTime": "2019-04-01 09:00",
    "ValidEndDateTime": "2019-04-30 23:59"
  },
  "report": "thing.event.UserValidDateTime.post",
}
```

```
    "type": "notification"
}
```

2.3.13. 获取锁用户的有效星期

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.GetUserValidWeekTime",
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid": <nodeid>,                   //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID>                  //锁用户 ID
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                      //1-成功 非 1-失败
  "errorinfo": "",                      //错误信息
  "errorinfo_cn": "",                  //本地化错误信息(Accept-Language 相关)
  "params": {
    "ValidWeekDays": <SumWeekdayValue>, //有效星期累加值: 如 5 表示星期天和星期二有效
                                           1-星期天 2-星期一 4-星期二 8-星期三
                                           16-星期四 32-星期五 64-星期六
    "ValidWeekTime": "<HH:MM~HH:MM>"     //有效星期的起止时间
  }
}
```

【注】

[示例]

```
curl -X GET \
  http(s)://dev.isurpass.com.cn:8888/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.GetUserValidWeekTime",
    "deviceid": "iRemote7005000000020",
    "nodeid": 123,
    "params": {
      "UserID": 11
    }
  }
,
```

//返回结果:

```
{
  "deviceid": "iRemote7005000000020",
  "id": 0,
```

```

    "logicnodeid": "iRemote7005000000020-123-0-",
    "nodeid": 123,
    "params": {
        "ValidWeekDays": 127,
        "ValidWeekTime": "00:00~23:59"
    },
    "report": "thing.event.UserValidWeekTime.post",
    "type": "notification"
}

```

2.3.14. 添加卡

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.AddCard",
    "async": false,                       //false-同步模式(缺省), true-异步模式
    "deviceid": "<deviceid>",             //网关设备 ID
    "nodeid": <nodeid>,                  //智能锁在网关设备上的节点 id
    "params": {
        "UserID": <UserID>               //锁用户 ID: 1~230
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,                      //1-成功 非 1-失败
    "receipt": "<receipt>",               //异步调用的回执 id
    "errorinfo": "",                      //错误信息
    "errorinfo_cn": "",                   //本地化错误信息(Accept-Language 相关)
    "params": {
        "UserType": <UserType>,          //1-密码 2-指纹 3-MF 卡 4-身份证
        "UserLevel": <UserLevel>,        //1-管理员 2-普通用户 4-一次性临时密码(UserType 必须为 1)
        "UserID": <UserID>,               //卡用户 ID
        "UserPassword": "<UserPassword>" //卡数据
    }
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1

[示例]

```

curl -X POST \
    http(s)://112.74.20.113:8888/lock \
    -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
    -H 'cache-control: no-cache' \
    -d '{

```

```

        "method": "thing.service.AddCard",
        "deviceid": "iSurpass2018100000",
        "nodeid": 2
    }
}
//返回结果:
{
    "params": {
        "UserID": 20,
        "UserLevel": 2,
        "UserPassword": "C7AF0BD6",
        "UserType": 3
    },
    "resultcode": 1
}

```

2.3.15. 添加指纹

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.AddFingerPrint",
    "async": false,                        //false-同步模式(缺省), true-异步模式
    "deviceid": "<deviceid>",             //网关设备 ID
    "nodeid": <nodeid>,                  //智能锁在网关设备上的节点 id
    "params": {
        "UserID": <UserID>               //锁用户 ID: 1~230
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,                      //1-成功 非 1-失败
    "receipt": "<receipt>",               //异步调用的回执 id
    "errorinfo": "",                      //错误信息
    "errorinfo_cn": "",                   //本地化错误信息(Accept-Language 相关)
    "params": {
        "UserType": <UserType>,          //1-密码 2-指纹 3-MF 卡 4-身份证
        "UserLevel": <UserLevel>,        //1-管理员 2-普通用户 4-一次性临时密码(UserType 必须为 1)
        "UserID": <UserID>                //指纹用户 ID
    }
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1

[示例]

```

curl -X POST \
    http(s)://112.74.20.113:8888/lock \
    -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \

```

```

-H 'cache-control: no-cache' \
-d '{
  "method": "thing.service.AddFingerPrint",
  "deviceid": "iSurpass2018100000",
  "nodeid": 2
}'
//返回结果:
{
  "params": {
    "UserID": 33,
    "UserLevel": 2,
    "UserType": 2
  },
  "resultcode": 1
}

```

2.3.16. 设置时区

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.SetNodeTimezone",
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid": <nodeid>,                   //智能锁在网关设备上的节点 id
  "params": {
    "Timezone": <timezone>              //时区: -23 ~ 23
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,      //1-成功 非 1-失败
}

```

[示例]

```

curl -X POST \
  http(s)://dev.isurpass.com.cn:8888/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.SetNodeTimezone",
    "deviceid": "iRemote2005000007242",
    "nodeid": 6,
    "params": {
      "Timezone": 8
    }
  }'

```

```
//返回结果:
{
  "RawData": "1E0800050001000100",
  "resultcode": 1
}
```

2.3.17. 停用锁用户

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.PauseUser",
  "async": false,                        //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid": <nodeid>,                   //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID>                  //锁用户 ID
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                       //1-成功 非 1-失败
  "receipt": "<receipt>",                 //异步调用的回执 id
  "errorinfo": "",                       //错误信息
  "errorinfo_cn": ""                     //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1
- 3) 该接口对蓝牙/二维码锁用户无效

2.3.18. 恢复锁用户

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.RestoreUser",
  "async": false,                        //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid": <nodeid>,                   //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID>                  //锁用户 ID
  }
}
```

```

    }
}
响应结果(Content-Type: application/json):
{
    "resultcode": 1,        //1-成功 非 1-失败
    "receipt": "<receipt>", //异步调用的回执 id
    "errorinfo": "",       //错误信息
    "errorinfo_cn": ""     //本地化错误信息(Accept-Language 相关)
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1
- 3) 该接口对蓝牙/二维码锁用户无效

2.4. 开关

http(s)://<ip>:<port>/switch

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.4.1. 获取开关的当前状态

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.property.get",
    "logicnodeid": "<logicnodeid>",      //开关逻辑节点 id
    "params": {
        "ForceRefresh": <true/false>    //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,        //1-成功 非 1-失败
    "errorinfo": "",       //错误信息
    "errorinfo_cn": "",    //本地化错误信息(Accept-Language 相关)
    "data": {
        "IsFault": <true/false>,      //开关当前是否是故障状态: true-故障状态 false-正常状态
        "Status": "<status>",          //开关当前状态: on-开状态 off-关状态
    }
}

```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/switch \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.property.get",
    "logicnodeid": "iRemote7005000000020-14-0-"
  }'
//返回结果:
{
  "resultcode": 1,
  "data": {
    "IsFault": false,
    "Status": "off"
  }
}
```

2.4.2. 远程开

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.Turnon",
  "logicnodeid": "<logicnodeid>" //开关逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/switch \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnon",
    "logicnodeid": "iRemote7005000000020-14-0-"
  }'
//返回结果:
{
  "resultcode": 1
```

```
}
```

2.4.3. 远程关

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.Turnoff",
  "logicnodeid": "<logicnodeid>" //开关逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/switch \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnoff",
    "logicnodeid": "iRemote7005000000020-14-0-"
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.5. 窗帘

http(s)://<ip>:<port>/curtain

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.5.1. 获取窗帘的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "ForceRefresh": "<true/false> //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": "<true/false>", //窗帘当前是否是故障状态: true-故障状态 false-正常状态
    "OpenPercentage": "<percent>", //窗帘打开的程度(百分比), 0 表示关状态
  }
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/curtain \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.property.get",
    "logicnodeid": "iRemote7005000000020-18-0-"
  }'
```

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "IsFault": false,
    "OpenPercentage": 50
  }
}
```

2.5.2. 远程开

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.Turnon",
  "logicnodeid": "<logicnodeid>" //逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/curtain \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnon",
    "logicnodeid": "iRemote7005000000020-18-0-"
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.5.3. 远程开(指定百分比)

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.set",
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "OpenPercentage": <percent> //打开的程度(百分比)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/curtain \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnon",
    "logicnodeid": "iRemote7005000000020-18-0-",
    "params": {
      "OpenPercentage": 30                //打开的程度(百分比)
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.5.4. 远程关

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.Turnoff",
  "logicnodeid": "<logicnodeid>" //逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,           //1-成功 非 1-失败
  "errorinfo": "",           //错误信息
  "errorinfo_cn": ""        //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/curtain \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnoff",
    "logicnodeid": "iRemote7005000000020-18-0-"
  }'
//返回结果:
{
  "resultcode": 1
}
```

```
}
```

2.6. 调光灯

http(s)://<ip>:<port>/dimmer

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.6.1. 获取调光灯的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "ForceRefresh": <true/false> //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": <true/false>, //调光灯当前是否是故障状态: true-故障状态 false-正常状态
    "OpenPercentage": <percent> //调光灯的亮度程度(百分比), 0 表示关状态
  }
}
```

[示例]

curl -X POST \

```
http(s)://112.74.20.113:8888/dimmer \
-H 'Content-Type: application/json' \
-H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
-d '{
  "method": "thing.service.property.get",
  "logicnodeid": "iRemote7005000000020-20-0-"
}'
```

//返回结果:

```
{
  "resultcode": 1,
```

```
    "data": {
      "IsFault": false,
      "OpenPercentage": 50
    }
  }
}
```

2.6.2. 远程开

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.Turnon",
  "logicnodeid": "<logicnodeid>" //逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/dimmer \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBBN-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnon",
    "logicnodeid": "iRemote7005000000020-20-0-"
  }'
```

//返回结果:

```
{
  "resultcode": 1
}
```

2.6.3. 远程开(指定亮度百分比)

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.property.set",
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "OpenPercentage": <percent> //打开的程度(百分比)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/dimmer \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnon",
    "logicnodeid": "iRemote7005000000020-20-0-",
    "params": {
      "OpenPercentage": 30 //打开的亮度程度(百分比)
    }
  }'
```

//返回结果:

```
{
  "resultcode": 1
}
```

2.6.4. 远程关

方法: post

请求头(header):

```
Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.Turnoff",
  "logicnodeid": "<logicnodeid>" //逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/dimmer \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslhRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnoff",
    "logicnodeid": "iRemote7005000000020-20-0-"
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.7. 门磁/窗磁

http(s)://<ip>:<port>/doorwindowsensor

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.7.1. 获取门磁/窗磁的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>" //逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": <true/false>, //门磁/窗磁当前是否是故障状态: true-故障状态 false-正常状态
    "BatteryPercentage": <percent>, //门磁/窗磁电量(百分比)
    "Opened": <true/false> //门磁/窗磁当前是否是打开状态: true-是 false-否
  }
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/doorwindowsensor \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.property.get",
    "logicnodeid": "iRemote7005000000020-50-0-"
  }'
//返回结果:
{
  "resultcode": 1,
  "data": {
    "BatteryPercentage": 100,
    "IsFault": false,
    "Opened": false
  }
}
```

2.8. 移动感应

http(s)://<ip>:<port>/motion

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.8.1. 获取移动感应的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>" //逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": <true/false>, //移动感应当前是否是故障状态: true-故障状态 false-正常状态
    "BatteryPercentage": <percent> //移动感应电量(百分比)
  }
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/motion \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.property.get",
    "logicnodeid": "iRemote7005000000020-50-0-"
  }'
//返回结果:
{
  "resultcode": 1,
  "data": {
    "BatteryPercentage": 100,
    "IsFault": false
  }
}
```

2.9. 空调

http(s)://<ip>:<port>/aircondition

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.9.1. 获取空调的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "ForceRefresh": <true/false> //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
```

```

    "IsFault": "<true/false>", //空调当前是否是故障状态: true-故障状态 false-正常状态
    "Status": "<on/off>", //空调当前开/关状态: on-开状态 off-关状态
    "Mode": "<mode>", //空调当前模式: auto-自动 hot-制热 cold-制冷 dry-抽湿 fan-送风
    "FanMode": "<fanmode>", //当前风速模式: auto-自动 low-低档 medium-中档 high-高档
    "PointTemperature-cold": { //制冷模式的设置温度
        "Scale": "<C/F>", //温度单位: C-摄氏度 F-华氏度
        "Value": "<value>" //温度值
    },
    "PointTemperature-hot": { //制热模式的设置温度
        "Scale": "<C/F>", //温度单位: C-摄氏度 F-华氏度
        "Value": "<value>" //温度值
    },
    "PointTemperature-dry": { //抽湿模式的设置温度
        "Scale": "<C/F>", //温度单位: C-摄氏度 F-华氏度
        "Value": "<value>" //温度值
    },
    "RoomTemperature": { //当前室温
        "Scale": "<C/F>", //温度单位: C-摄氏度 F-华氏度
        "Value": "<value>" //温度值
    },
    "BatteryPercentage": "<percent>", //电量(百分比)
}
}

```

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/aircondition \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcCI6ImlRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.property.get",
    "logicnodeid": "iRemote7005000000020-61-0-"
  }'

```

//返回结果:

```

{
  "resultcode": 1,
  "data": {
    "FanMode": "high",
    "IsFault": false,
    "Mode": "hot",
    "PointTemperature-cold": {
      "Scale": "C",
      "Value": 20
    },
    "PointTemperature-hot": {
      "Scale": "C",
      "Value": 26
    },
    "RoomTemperature": {
      "Scale": "C",
      "Value": 24.3
    },
    "Status": "on"
  }
}

```

2.9.2. 远程开

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.Turnon",
  "logicnodeid": "<logicnodeid>" //逻辑节点 id
}
或者:
{
  "method": "thing.service.Turnon",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //在网关设备上的节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/aircondition \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnon",
    "logicnodeid": "iRemote7005000000020-61-0-"
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.9.3. 远程关

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.Turnoff",
```

```

    "logicnodeid": "<logicnodeid>"          //逻辑节点 id
}
或者:
{
    "method": "thing.service.Turnoff ",
    "deviceid": "<deviceid>",              //网关设备 ID
    "nodeid": "<nodeid>",                  //在网关设备上的节点 id
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,          //1-成功 非 1-失败
    "errorinfo": "",          //错误信息
    "errorinfo_cn": ""       //本地化错误信息(Accept-Language 相关)
}

```

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/aircondition \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrczkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnoff",
    "logicnodeid": "iRemote7005000000020-61-0-"
  }'
//返回结果:
{
    "resultcode": 1
}

```

2.9.4. 远程设置空调运行模式

方法: post

请求头(header):

```

Accept-Language: <language>          //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.SetMode",
    "logicnodeid": "<logicnodeid>",      //逻辑节点 id
    "params": {
        "Mode": "<Mode>"                // auto-自动 hot-制热 cold-制冷 dry-抽湿 fan-送风
    }
}
或者:
{
    "method": "thing.service.SetMode ",
    "deviceid": "<deviceid>",            //网关设备 ID
    "nodeid": "<nodeid>",                //在网关设备上的节点 id
    "params": {
        "Mode": "<Mode>"                // auto-自动 hot-制热 cold-制冷 dry-抽湿 fan-送风
    }
}

```

```
}  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1,      //1-成功 非 1-失败  
  "errorinfo": "",      //错误信息  
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)  
}
```

[示例]

```
curl -X POST \  
  http(s)://112.74.20.113:8888/aircondition \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization:Bearer  
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \  
  -d '{  
    "method": "thing.service.SetMode",  
    "logicnodeid": "iRemote7005000000020-61-0-",  
    "params": {  
      "Mode": "hot"  
    }  
  }'  
//返回结果:  
{  
  "resultcode": 1  
}
```

2.9.5. 远程设置空调温度

方法: post

请求头(header):

```
Accept-Language: <language>      //en、zh-CN  
Authorization: Bearer <JWT token>
```

请求参数:

```
{  
  "method": "thing.service.SetPointTemperature",  
  "logicnodeid": "<logicnodeid>",    //逻辑节点 id  
  "params": {  
    "Mode": "<Mode>",                //模式: hot-制热 cold-制冷 dry-抽湿  
    "Value": <value>,                //温度值  
    "Scale": "<C/F>"                 //温度单位: C-摄氏度 F-华氏度  
  }  
}
```

或者:

```
{  
  "method": "thing.service.SetPointTemperature",  
  "deviceid": "<deviceid>",          //网关设备 ID  
  "nodeid": <nodeid>,                //在网关设备上的节点 id  
  "params": {  
    "Mode": "<Mode>",                // hot-制热 cold-制冷 dry-抽湿  
    "Value": <value>,                //温度值  
  }  
}
```

```

        "Scale": "<C/F>" //温度单位: C-摄氏度 F-华氏度
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/aircondition \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.SetPointTemperature",
    "logicnodeid": "iRemote700500000020-61-0-",
    "params": {
      "Mode": "hot",
      "Value": 26,
      "Scale": "C"
    }
  }'
//返回结果:
{
  "resultcode": 1
}

```

2.9.6. 远程设置空调风速

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.SetFanMode",
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "params": {
        "FanMode": "<FanMode>" // auto-自动 low-低档 medium-中档 high-高档
    }
}

```

或者:

```

{
    "method": "thing.service.SetFanMode",
    "deviceid": "<deviceid>", //网关设备 ID
    "nodeid": "<nodeid>", //在网关设备上的节点 id
    "params": {
        "FanMode": "<FanMode>" // auto-自动 low-低档 medium-中档 high-高档
    }
}

```

```
}  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1,      //1-成功 非 1-失败  
  "errorinfo": "",      //错误信息  
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)  
}
```

[示例]

```
curl -X POST \  
  http(s)://112.74.20.113:8888/aircondition \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization:Bearer  
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \  
  -d '{  
    "method": "thing.service.SetFanMode",  
    "logicnodeid": "iRemote7005000000020-61-0-",  
    "params": {  
      "FanMode": "low"  
    }  
  }'  
//返回结果:
```

```
{  
  "resultcode": 1  
}
```

2.10. 温度计

http(s)://<ip>:<port>/temperaturesensor

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.10.1. 获取温度计的当前状态

方法: post

请求头(header):

```
Accept-Language: <language>      //en、zh-CN  
Authorization: Bearer <JWT token>
```

请求参数:

```
{  
  "method": "thing.service.property.get",  
  "logicnodeid": "<logicnodeid>"    //逻辑节点 id  
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",      //错误信息
  "errorinfo_cn": "",   //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": <true/false>, //温度计当前是否是故障状态: true-故障状态 false-正常状态
    "BatteryPercentage": <percent>, //温度计电量(百分比)
    "Temperature": {
      "Scale": "<C/F>", //温度单位: C-摄氏度 F-华氏度
      "Value": <value> //温度值
    }
  }
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/temperaturesensor \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.property.get",
    "logicnodeid": "iRemote7005000000020-98-0-temperaturesensor"
  }'
```

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "IsFault": false,
    "Temperature": {
      "Scale": "C",
      "Value": 25.4
    }
  }
}
```

2.11. 亮度计

http(s)://<ip>:<port>/luminancesensor

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.11.1. 获取亮度计的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.property.get",  
  "logicnodeid": "<logicnodeid>" //逻辑节点 id  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "data": {  
    "IsFault": <true/false>, //亮度计当前是否是故障状态: true-故障状态 false-正常状态  
    "BatteryPercentage": <percent>, //亮度计电量(百分比)  
    "Luminance": {  
      "Scale": "<%/Lux>", //亮度单位: %-百分比 Lux-勒克斯(照明单位)  
      "Value": <value> //亮度值  
    }  
  }  
}
```

[示例]

```
curl -X POST \  
  http(s)://112.74.20.113:8888/luminancesensor \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization:Bearer  
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \  
  -d '{  
    "method": "thing.service.property.get",  
    "logicnodeid": "iRemote7005000000020-98-0-luminancesensor"  
  }'
```

//返回结果:

```
{  
  "resultcode": 1,  
  "data": {  
    "IsFault": true,  
    "Luminance": {  
      "Scale": "%",  
      "Value": 9  
    }  
  }  
}
```

2.12. 湿度计

http(s)://<ip>:<port>/humiditysensor

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.12.1. 获取湿度计的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.property.get",  
  "logicnodeid": "<logicnodeid>" //逻辑节点 id  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "data": {  
    "IsFault": <true/false>, //湿度计当前是否是故障状态: true-故障状态 false-正常状态  
    "BatteryPercentage": <percent>, //湿度计电量(百分比)  
    "Humidity": {  
      "Scale": "<%/g/m3>", //湿度单位: %-百分比 g/m3-绝对湿度  
      "Value": <value> //湿度值  
    }  
  }  
}
```

[示例]

```
curl -X POST \  
  http(s)://112.74.20.113:8888/humiditysensor \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \  
  -d {  
    "method": "thing.service.property.get",  
    "logicnodeid": "iRemote7005000000020-98-0-humiditysensor"  
  }  
}
```

//返回结果:

```
{  
  "resultcode": 1,  
  "data": {  
    "IsFault": true,  
    "Humidity": {  
      "Scale": "%",  
      "Value": 65  
    }  
  }  
}
```

2.13. 烟感

http(s)://<ip>:<port>/smokesensor

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.13.1. 获取烟感的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>" //逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": <true/false>, //设备当前是否是故障状态: true-故障状态 false-正常状态
    "BatteryPercentage": <percent> //设备电量(百分比)
  }
}
```

[示例]

curl -X POST \

http(s)://112.74.20.113:8888/smokesensor \

-H 'Accept: */*' \

-H 'Authorization: Bearer

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \

-d '{

"method": "thing.service.property.get",

"logicnodeid": "iRemote7005000000020-130-0-"

}'

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "BatteryPercentage": 100,
    "IsFault": false
  }
}
```

2.14. 漏水感应

http(s)://<ip>:<port>/watersensor

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.14.1. 获取漏水感应的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.property.get",  
  "logicnodeid": "<logicnodeid>" //逻辑节点 id  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "data": {  
    "IsFault": <true/false>, //设备当前是否是故障状态: true-故障状态 false-正常状态  
    "BatteryPercentage": <percent> //设备电量(百分比)  
  }  
}
```

[示例]

curl -X POST \

http(s)://112.74.20.113:8888/watersensor \

-H 'Accept: */*' \

-H 'Authorization: Bearer

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \

-d '{

"method": "thing.service.property.get",

"logicnodeid": "iRemote7005000000020-11-0-"

}'

//返回结果:

```
{  
  "resultcode": 1,  
  "data": {  
    "BatteryPercentage": 100,  
    "IsFault": false  
  }  
}
```

2.15. 红外控制

http(s)://<ip>:<port>/irremote

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.15.1. 获取红外控制的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>", //红外控制逻辑节点 id
  "params": {
    "ForceRefresh": <true/false> //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": <true/false>, //红外控制当前是否是故障状态: true-故障状态 false-正常状态
    "Status": "<status>", //红外控制当前状态: on-开状态 off-关状态
  }
}
```

[示例]

curl -X POST \

```
http(s)://112.74.20.113:8888/irremote \
-H 'Content-Type: application/json' \
-H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBBN-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
-d '{
  "method": "thing.service.property.get",
  "logicnodeid": "iRemote7005000000020-9-0-"
}'
```

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "IsFault": false,
```

} }

请求头(head)

Accept-Language

Authorization: Bearer <JWT token>

求参数:

$$\{$$

```

"method": "thing.service.IRLearn",
"logicnodeid": "<logicnodeid>" //红外控制逻辑节点 id
}

```

$$\{$$

```
"resultCode": 1, //1-成功 非 1-失败
"errorinfo": "", //错误信息
"errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
"params": {
    "IRData": "< hex>" //红外控制指令数据, HEX 格式
}
```

```
curl -X
```

```
http(s)://112.74.20.113:8888/irremote \
-H 'Content-Type: application/json' \
-H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBBrZXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8B8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
-d '{
  "method": "thing.service.IRLearn",
  "logicnodeid": "iRemote7005000000020-9-0-"
}
```

$$\{$$
[illegible]

方法: post

请求头(header):

```
Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.Turnoff",
  "logicnodeid": "<logicnodeid>", //红外控制逻辑节点 id
  "params": {
    "IRData": "<hex>" //红外控制指令数据, HEX 格式
  }
}
```

响应结果(Content-Type: application/json):

```
{
    "resultCode": 1,           //1-成功 非 1-失败
    "errorinfo": "",          //错误信息
    "errorinfo_cn": ""        //本地化错误信息(Accept-Language 相关)
}
```

[示例]

[illegible]

//返回结果:

```
{
  "resultCode": 1
}
```

2.15.5. 远程控制

方法: post

请求头(header):

```
Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.IRControl",

```

响应结果(Content-Type: application/json):

[示例]

2.16. 电表

2.16.1. 获取电表的当前状态

方法: post
请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.property.get",  
  "logicnodeid": "<logicnodeid>" //电表逻辑节点 id  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "data": {  
    "IsFault": <true/false>, //电表当前是否是故障状态: true-故障状态 false-正常状态  
    "Kwh": <kwh>, //电表当前读数  
  }  
}
```

[示例]

```
curl -X POST \  
  http(s)://112.74.20.113:8888/electricmeter \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization:Bearer  
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcGlzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \  
  -d '{  
    "method": "thing.service.property.get",  
    "logicnodeid": "onenet333333333333-1-0-"  
  }'  
//返回结果:  
{  
  "resultcode": 1,  
  "data": {  
    "IsFault": false,  
    "KWh": "0.19"  
  }  
}
```

2.17. 水表

http(s)://<ip>:<port>/watermeter

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.17.1. 获取水表的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.property.get",  
  "logicnodeid": "<logicnodeid>" //水表逻辑节点 id  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "data": {  
    "IsFault": <true/false>, //水表当前是否是故障状态: true-故障状态 false-正常状态  
    "CubicMeters": <cubic meters>, //水表当前读数  
  }  
}
```

[示例]

```
curl -X POST \  
  http(s)://112.74.20.113:8888/watermeter \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization:Bearer  
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \  
  -d '{  
    "method": "thing.service.property.get",  
    "logicnodeid": "onenet203600020619-1-0-"  
  }'
```

//返回结果:

```
{  
  "resultcode": 1,  
  "data": {  
    "CubicMeters": "0.7",  
    "IsFault": false  
  }  
}
```

2.18. 人脸识别

2.19. AV 音视频设备

http(s)://<ip>:<port>/av

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.19.1. 获取音视频设备的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>", //红外控制逻辑节点 id
  "params": {
    "ForceRefresh": "<true/false> //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": "<true/false>", //AV 设备当前是否是故障状态: true-故障状态 false-正常状态
    "Status": "<on/off>", //AV 设备当前状态: on-开状态 off-关状态
    "IsMute": "<true/false>", //当前是否静音
    "PlayMode": "<play mode>", //当前播放模式
    "PlayStatus": "<play status>", //当前播放状态
    "Volume": "<volume>" //当前音量
  }
}
```

2.19.2. 远程开

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.Turnon",
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

2.19.3. 远程关

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.Turnoff",  
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)  
}
```

2.19.4. 远程设置音量

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.SetVolume",  
  "logicnodeid": "<logicnodeid>", //设备的逻辑节点 id  
  "params": {  
    "Volume": <volume> //音量: 0~100  
  }  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)  
}
```

2.19.5. 远程调低/高音量

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.VolumeDown",
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id
}
或
{
  "method": "thing.service.VolumeUp",
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

2.19.6. 远程设置静音/取消静音

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.Mute",
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id
}
或
{
  "method": "thing.service.UnMute",
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

2.19.7. 远程控制暂停/恢复播放/停止播放

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.Pause",
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id
}
或
{
  "method": "thing.service.Play",
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id
}
或
{
  "method": "thing.service.Stop",
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

2.19.8. 远程控制播放下一首/上一首

方法: post

请求头(header):

```
Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.Next",
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id
}
或
{
  "method": "thing.service.Previous",
  "logicnodeid": "<logicnodeid>" //设备的逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

2.19.9. 远程设置播放模式

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetPlayMode",
  "logicnodeid": "<logicnodeid>", //设备的逻辑节点 id
  "params": {
    "Mode": "<allcycle/randomcycle/singlecycle>" //播放模式
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

2.20. 中控多媒体主机

http(s)://<ip>:<port>/pcmedia

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.20.1. 远程获取视频文件列表

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetVideoList",
  "deviceid": "<deviceid>", //TCP 网关设备 ID
  "nodeid": <nodeid>, //多媒体主机在 TCP 网关上的节点 id
  "params": {
    "port": <port> //多媒体主机的文件列表服务端口
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "list": [<filename>] //文件名数组
}
```

```
}
```

2.20.2. 远程打开视频文件

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.OpenVideoFile",
  "deviceid": "<deviceid>", //TCP 网关设备 ID
  "nodeid": <nodeid>, //多媒体主机在 TCP 网关上的节点 id
  "params": {
    "port": <port>, //多媒体主机的文件列表服务端口
    "filename": "<filename>" //视频文件名
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

2.20.3. 远程获取 PPT 文件列表

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetPPTList",
  "deviceid": "<deviceid>", //TCP 网关设备 ID
  "nodeid": <nodeid>, //多媒体主机在 TCP 网关上的节点 id
  "params": {
    "port": <port> //多媒体主机的文件列表服务端口
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "list": [<filename>] //文件名数组
}
```

2.20.4. 远程打开 PPT 文件

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.OpenPPTFile",
  "deviceid": "<deviceid>", //TCP 网关设备 ID
  "nodeid": "<nodeid>", //多媒体主机在 TCP 网关上的节点 id
  "params": {
    "port": <port>, //多媒体主机的文件列表服务端口
    "filename": "<filename>" //PPT 文件名
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

2.20.5. 远程热键控制

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.HotkeyControl",
  "deviceid": "<deviceid>", //TCP 网关设备 ID
  "nodeid": "<nodeid>", //多媒体主机在 TCP 网关上的节点 id
  "params": {
    "port": <port>, //多媒体主机的远程热键服务端口
    "hotkey": "<hotkey>" //热键
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

2.21. 净水器

http(s)://<ip>:<port>/waterfilter

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.21.1. 获取净水器的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>" //净水器逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": <true/false>, //设备当前是否是故障状态: true-故障状态 false-正常状态
    "CardNumber": "<card number>", //NB 卡卡号
    "Model": "<value>", //净水器型号

    "InWaterQuality": <value>, //入水水质
    "OutWaterQuality": <value>, //出水水质
    "MakeWaterStatus": <value>, //制水状态
    "MakeWaterTime": <value>, //制水时长
    "MakeWaterTotal": <value>, //制水量
    "RemainingTimes": <value> //滤芯剩余时间
  }
}
```

2.22. 滚筒洗衣机

http(s)://<ip>:<port>/washer

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.22.1. 获取洗衣机的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.property.get",  
  "deviceid": "<deviceid>", //网关设备 ID  
  "nodeid": "<nodeid>" //洗衣机在网关上的节点 id, 固定值 1  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "data": {  
    "IsFault": <true/false>, //设备当前是否是故障状态: true-故障状态 false-正常状态  
    "Online": <true/false>, //设备当前是否在线: true-在线 false-离线  
    "CardNumber": "<card number>", //NB 卡卡号  
    "RSSI": <RSSI>, //NB 信号强度  
  
    "Power": <power>, //电源: 0-关闭 1-开启  
    "Mode": <mode>, //模式: 0-不设定 1-单脱水 5-快洗 8-标准洗 10-大物洗 12-加温洗  
    35-筒自洁 36-高温杀菌洗 37-羽绒服 38-洗鞋  
    "Washings": <value>, //洗涤液液位: 0-告警 1-正常  
    "DryTime": <time>, //烘干时间: 0-不设定 16-15 分钟 32-30 分钟 48-60 分钟  
    64-90 分钟 80-120 分钟 96-150 分钟  
    "DryMode": <mode>, //烘干模式: 0-不设定 1-标准 2-特干 3-晾干 4-取消  
    "SerialFault": <value>, //串口故障: 0-正常 32-数据错乱 64-丢包  
    "WasherFault": <value>, //洗衣机故障: 0-正常 1-进水超时 2-排水超时 3-溢水 4-水位传感  
    5-温度加热器 6-门锁 7-脱水平衡 8-外部存储 9-过电流 10-位置传感  
    11-直流电压 12-烘干加热器 13-微安全  
    "RemainingTime": <value>, //洗衣剩余时间, 单位分钟  
    "ProgramTime": <value>, //程序总时间, 单位分钟  
    "ProductModel": "<model>", //洗衣机型号  
    "DryingTime": <value> //烘干加热时间, 单位秒  
  }  
}
```

[示例]

curl -X POST \

http(s)://dev.isurpass.com.cn:8888/washer \

-H 'Content-Type: application/json' \

-H 'Authorization: Bearer

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBBN-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \

-d '{

"method": "thing.service.property.get",

"deviceid": "iSurpass2020020202",

"nodeid": 1

}'

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "DryMode": 1,
    "DryTime": 48,
    "DryingTime": 0,
    "IsFault": false,
    "Mode": 5,
    "Power": 1,
    "ProductModel": "01",
    "ProgramTime": 60,
    "RemainingTime": 20,
    "SerialFault": 0,
    "WasherFault": 0,
    "Washings": 1
  }
}
```

2.22.2. (NB-IoT)远程控制

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.Control",
  "async": true,           //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>",    //洗衣机控制设备在网关上的节点 id
  "params": {
    "Power": <power>,      //电源: 0-关闭 1-开启
    "Mode": <mode>,        //模式: 0-不设定 1-单脱水 5-快洗 8-标准洗 10-大物洗 12-加温洗
                           35-筒自洁 36-高温杀菌洗 37-羽绒服 38-洗鞋
    "DropWashingsTime": <time>, //洗涤剂投放时间, 单位 100ms
    "DryTime": <time>,        //烘干时间: 0-不设定 16-15 分钟 32-30 分钟 48-60 分钟
                           64-90 分钟 80-120 分钟 96-150 分钟
    "DryMode": <mode>,       //烘干模式: 0-不设定 1-标准 2-特干 3-晾干 4-取消
    "Mode1Time": <time>,     //单脱水时间: 5~9 分钟
    "Mode5Time": <time>,     //快洗时间: 25~32 分钟
    "Mode8Time": <time>,     //标准洗时间: 35~42 分钟
    "Mode10Time": <time>,    //大物洗时间: 45~52 分钟
    "Mode12Time": <time>     //加温洗时间: 55~60 分钟
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) ~~async 参数值必须为 true~~, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://dev.isurpass.com.cn:8888/washer \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "async": true,
    "method": "thing.service.Control",
    "deviceid": "iSurpass2020020202",
    "nodeid": 1,
    "params": {
      "Power": 1,
      "Mode": 5,
      "DropWashingsTime": 10,
      "DryTime": 32,
      "DryMode": 1,
      "Mode1Time": 5,
      "Mode5Time": 25,
      "Mode8Time": 35,
      "Mode10Time": 45,
      "Mode12Time": 60
    }
  }'
//返回结果:
{
  "resultcode": 1,
  "receipt": "washer-1580721877006"
}
```

2.22.3. (NB-IoT)远程复位

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.Reset",
  "async": true,           //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>       //洗衣机控制设备在网关上的节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,        //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "",         //错误信息
  "errorinfo_cn": ""       //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 `async` 参数为 `true` 时, `resultcode` 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 `websocket` 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) ~~`async` 参数值必须为 `true`~~, `nodeid` 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://dev.isurpass.com.cn:8888/washer \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "async": true,
    "method": "thing.service.Reset",
    "deviceid": "iSurpass2020020202",
    "nodeid":1
  }'
//返回结果:
{
  "resultcode": 1,
  "receipt": "washer-1580721868139"
}
```

2.23. 波轮洗衣机

`http(s)://<ip>:<port>/washer`

encoding: UTF-8

*开发环境可以用 `http`, 生产环境建议使用 `https`

2.23.1. 获取洗衣机的当前状态

方法: `post`

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.property.get",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>       //洗衣机在网关上的节点 id, 固定值 1
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
```

```

"data": {
  "IsFault": <true/false>,    //设备当前是否是故障状态: true-故障状态 false-正常状态
  "Online": <true/false>,    //设备当前是否在线: true-在线 false-离线
  "CardNumber": "<card number>",    //NB 卡卡号
  "RSSI": <RSSI>,            //NB 信号强度

  "Status": <status>,        //状态: 0-待机 1-预约中 2-消毒中 3-洗衣中 4-洗涤结束 5-自检中
                                6-进水超时 7-排水超时 8-脱水撞桶 9-脱水开盖 10-水位传感异常
                                11-溢水报警 12-电机故障
  "RemainingTime": <value>, //剩余时间
  "Coins": <count>           //投币数
}

```

2.23.2. (NB-IoT)远程控制

方法: post

请求头(header):

```

Accept-Language: <language>    //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.Control",
  "async": true,            //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>,      //洗衣机控制设备在网关上的节点 id
  "params": {
    "Cmd": <cmd>,          //命令: 1-未定义 2-工作 3-预约 5-取消预约
    "Mode": <mode>         //模式: 1-加强洗 2-标准洗 3-快速洗 4-单脱水 5-桶消毒 6-自检
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,    //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "",    //错误信息
  "errorinfo_cn": ""  //本地化错误信息(Accept-Language 相关)
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) ~~async 参数值必须为 true~~, nodeid 参数值必须为 1

2.24. 波轮洗衣机 B

http(s)://<ip>:<port>/washer

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.24.1. 获取洗衣机的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.property.get",  
  "deviceid": "<deviceid>", //网关设备 ID  
  "nodeid": "<nodeid>" //洗衣机在网关上的节点 id, 固定值 1  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "data": {  
    "IsFault": <true/false>, //设备当前是否是故障状态: true-故障状态 false-正常状态  
    "Online": <true/false>, //设备当前是否在线: true-在线 false-离线  
    "CardNumber": "<card number>", //NB 卡卡号  
    "RSSI": <RSSI>, //NB 信号强度  
  
    "Mode": <mode>, //当前运行模式: 0: 无  
    1: 标准洗  
    2: 快速洗  
    3: 强力洗  
    4: 单脱水  
    5: [轻柔洗]  
    6: [漂洗并脱水]  
    7: 桶清洁  
    8: [能效-全载]  
    9: [能效-半载]  
    10: [整机自检]  
    11: [电脑板自检]  
  
    "Status": <status>, //当前洗衣状态: 0: [预留]  
    1: [预留]  
    2: [预留]  
    3: 洗涤中  
    4: 漂洗中  
    5: 脱水中  
    6: 洗衣结束  
    7: 异常  
  
    "RemainingTime": <value>, //剩余时间  
    "WaterLevel": <water level>, //洗涤水位: 1~8 档  
    "ErrorStatus": "<hex value>" //异常状态(十六进制串)  
    00: 无出错;  
    01: 进水超时(故障代码: E0)  
    02: 排水超时(故障代码: E1)  
    03: 脱水不平衡(故障代码: E2)
```

```

04: 门开关报警(故障代码: E3)
05: 水位传感器异常(故障代码: E4)
06: 预约开门(故障代码: E5)
07: 电机故障报警(故障代码: E6)
08: 童锁开门(故障代码: E7)
09: 存储器故障(故障代码: E8)
}

```

2.24.2. (NB-IoT)远程控制

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.Control",
  "async": true,           //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>,      //洗衣机控制设备在网关上的节点 id
  "params": {
    "Cmd": <cmd>,          //命令: 0: 无
                           1: 标准洗
                           2: 快速洗
                           3: 强力洗
                           4: 单脱水
                           5: [轻柔洗]
                           6: [漂洗并脱水]
                           7: 桶清洁1
                           8: [能效-全载]
                           9: [能效-半载]
                           10: [整机自检]
                           11: [电脑板自检]
    "StartKey": <status val>, //启动键状态: 0: 未启动 1: 启动洗衣 2: 停止洗衣
    "ExtraTime": <minutes>,   //洗衣结束后的占用时间: 1~20(分钟)
    "WaterLevel": <water level> //洗涤水位: 1~8 档
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) async 参数值必须为 true, nodeid 参数值必须为 1

2.25. 洗鞋机

http(s)://<ip>:<port>/shoewasher

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.25.1. 获取洗鞋机的当前状态&参数

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.property.get",  
  "deviceid": "<deviceid>", //网关设备 ID  
  "nodeid": "<nodeid>" //洗衣机在网关上的节点 id, 固定值 1  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "data": {  
    "IsFault": <true/false>, //设备当前是否是故障状态: true-故障状态 false-正常状态  
    "Online": <true/false>, //设备当前是否在线: true-在线 false-离线  
    "CardNumber": "<card number>", //NB 卡卡号  
    "RSSI": <RSSI>, //NB 信号强度  
  
    "Status": <status>, //运行状态 1-待机 2-运转 3-暂停 4-故障 5-参数设置 6-自检  
    "Program": <program>, //运行程序: 1-单脱水 2-快速 3-标准 4-大物  
    //5-加热快速 6-加热标准 7-加热大物  
    "Progress": <progress>, //运行过程: 1-预约 2-浸泡 3-洗涤 4-漂洗 5-脱水 6-主进水  
    "RemainingTime": <time>, //剩余时间(秒)  
    "Faults": <fault value>, //故障标识: 1-进水超时 2-排水超时 4-脱水开盖 8-脱水不平衡  
    //16-通讯故障 32-童锁开盖 64-溢水 128-水位传感故障  
    //256-数据存储故障 512-温度传感故障 1024-无水加热故障  
    //2048-电机堵转故障 4096-温度过高故障 8192-电机过流故障  
    //16384-电机缺相故障 32768-驱动器故障  
    "VersionNum": <version>, //版本号  
    "Features": <feature value>, //机型标识: 1-滚筒 2-杀菌灯 4-加液 8-水位可调 16-转速可调 32-加速  
    "MobilePay": <amount>, //手机支付钱数  
    "Coins": <count>, //投币数量  
  
    "Price3": <price>, //单脱价格  
    "Price4": <price>, //快速价格  
    "Price5": <price>, //标准价格  
    "Price6": <price>, //大物价格  
    "Price7": <price>, //加热快速价格
```

```

    "Price8": <price>, //加热标准价格
    "Price9": <price>, //加热大物价格

    "Time3": <time>, //单脱时间
    "Time4": <time>, //快速时间
    "Time5": <time>, //标准时间
    "Time6": <time>, //大物时间
    "Time7": <time>, //液泵时间
    "Time8": <time>, //加热快速时间
    "Time9": <time>, //加热标准时间
    "Time10": <time>, //加热大物时间

    "Parameter3": <value>, //快速水位设定
    "Parameter4": <value>, //标准水位设定
    "Parameter5": <value>, //大物水位设定
    "Parameter6": <value>, //脱水转速设定
    "Parameter7": <value>, //温度设定

    "SpecialOptions": <value>, //1-掉电记忆 2-全程保护 4-加热 8-桶清洁 16-按键 32-投币
  }
}

```

2.25.2. (NB-IoT)远程控制

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.Control",
  "async": true, //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //洗鞋机控制设备在网关上的节点 id
  "params": {
    "Mode1": <mode>, //模式: 1-大物洗 2-标准洗 4-快洗 8-单脱水 16-桶清洁 32-结束
    //64-暂停 128-恢复
    "Mode2": <mode> //模式: 0-全关闭 1-自检模式 1 开启 2-自检模式 2 开启 4-跳转开启 8-结束开启
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) async 参数值必须为 true, nodeid 参数值必须为 1

2.25.3. (NB-IoT)远程设定价格

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetPrice",
  "async": true,           //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>,      //洗鞋机控制设备在网关上的节点 id
  "params": {
    "Price3": <price>,      //单脱价格
    "Price4": <price>,      //快速价格
    "Price5": <price>,      //标准价格
    "Price6": <price>,      //大物价格
    "Price7": <price>,      //加热快速价格
    "Price8": <price>,      //加热标准价格
    "Price9": <price>       //加热大物价格
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) ~~async 参数值必须为 true~~, nodeid 参数值必须为 1

2.25.4. (NB-IoT)远程设定时间参数

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetTimeParameters",
  "async": true,           //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>,      //洗鞋机控制设备在网关上的节点 id
  "params": {
    "Time3": <time>,        //单脱时间
    "Time4": <time>,        //快速时间
    "Time5": <time>,        //标准时间
  }
}
```

```

        "Time6": <time>,           //大物时间
        "Time7": <time>,           //液泵时间
        "Time8": <time>,           //加热快速时间
        "Time9": <time>,           //加热标准时间
        "Time10": <time>           //加热大物时间
    }
}
响应结果(Content-Type: application/json):
{
    "resultcode": 1,    //1-成功 非 1-失败
    "receipt": "<receipt>", //异步调用的回执 id
    "errorinfo": "",    //错误信息
    "errorinfo_cn": ""  //本地化错误信息(Accept-Language 相关)
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) ~~async 参数值必须为 true~~, nodeid 参数值必须为 1

2.25.5. (NB-IoT)远程设定过程参数

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.SetProgramParameters",
    "async": true,           //false-同步模式(缺省), true-异步模式
    "deviceid": "<deviceid>", //网关设备 ID
    "nodeid": <nodeid>,     //洗鞋机控制设备在网关上的节点 id
    "params": {
        "Parameter3": <value>, //快速水位设定
        "Parameter4": <value>, //标准水位设定
        "Parameter5": <value>, //大物水位设定
        "Parameter6": <value>, //脱水转速设定
        "Parameter7": <value>  //温度设定
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,    //1-成功 非 1-失败
    "receipt": "<receipt>", //异步调用的回执 id
    "errorinfo": "",    //错误信息
    "errorinfo_cn": ""  //本地化错误信息(Accept-Language 相关)
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) ~~async 参数值必须为 true~~, nodeid 参数值必须为 1

2.25.6. (NB-IoT)远程设定特殊功能

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetSpecialOptions",
  "async": true,           //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>,      //洗鞋机控制设备在网关上的节点 id
  "params": {
    "SpecialOptions": <value>, //1-掉电记忆 2-全程保护 4-加热 8-桶清洁 16-按键 32-投币
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) ~~async 参数值必须为 true~~, nodeid 参数值必须为 1

2.26. 干衣机

http(s)://<ip>:<port>/dryer

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.26.1. 获取干衣机的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
```

```
"method": "thing.service.property.get",
"deviceid": "<deviceid>", //网关设备 ID
"nodeid": <nodeid>      //干衣机在网关上的节点 id, 固定值 1
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": <true/false>, //设备当前是否是故障状态: true-故障状态 false-正常状态
    "Online": <true/false>, //设备当前是否在线: true-在线 false-离线
    "CardNumber": "<card number>", //NB 卡卡号
    "RSSI": <RSSI>, //NB 信号强度

    "Status": <status>, //运行状态, 整数值
    "Status2": "<hex>", //特殊状态指示, 十六进制串
    "RemainingTime": <time>, //剩余时间(分钟)
    "Temperature": <val>, //温度
    "Humidity": <val>, //湿度
    "Version": "<hex>" //程序版本
  }
}
```

运行状态：

占用字节：1

对应数值	发送指令含义	备注
00H	空闲	可以使用
01H	准备中	不可使用
02H	正常工作中	不可使用
03H	工作结束	维持 3 秒
05H	上传温湿度值	
10H	E1	温度传感器失效
11H	E2	湿度传感器失效
12H	E3	温度湿度传感器同时故障
13H	E4	工作状态时开门被打开
23H	E5	待机时门打开
80H	桶清洁工作中	不可使用
81H	掉电自动工作中	服务器下发停止指令后，记忆状态清除， 进入停止状态

特殊状态指示：

占用字节：1

对应数值	发送指令含义	备注
Bit0 = 0	按键未锁	按键可用于测试工作
Bit0 = 1	按键已锁	按键不可用
Bit1 = 0	不启动智能判断	未达预置时间，衣干即停
Bit1 = 1	启动智能判断	按预置时间工作

2.26.2. (NB-IoT)远程控制

方法: post

请求头(header)：

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数：

```
{
  "method": "thing.service.Control",
  "async": true,                //false-同步模式(缺省), true-异步模式
  "deviceId": "<deviceId>",      //网关设备 ID
  "nodeid": "<nodeid>",          //干衣机控制设备在网关上的节点 id, 固定值 1
  "params": {
    "Cmd": "<hex>",              //命令, 十六进制串
  }
}
```

```

    "Select": "<hex>"          //干衣选择, 十六进制串
}
}
//命令:

```

01H	启动运行	启动干衣机运行
02H	停止运行	停止干衣机工作
03H	开放按键	按键可以使用
04H	封锁按键	按键禁止使用
05H	温湿度查询	读取干衣机当前温湿度值
06H	桶自洁启动运行	桶自洁启动功能
07H	不启用智能判断	不启动智能模式, 按预置时间工作
08H	启用智能判断	自动判断干衣, 衣干即停

//干衣选择:

对应数值	发送数据含义
00H	干衣时间 30 分钟
01H	干衣时间 35 分钟
02H	干衣时间 40 分钟
...	...
4FH	干衣时间 425 分钟
50H	干衣时间 5 分钟
51H	干衣时间 10 分钟
52H	干衣时间 15 分钟
53H	干衣时间 20 分钟
54H	干衣时间 25 分钟
80H	干衣时间 30 分钟 + 杀菌
81H	干衣时间 35 分钟 + 杀菌
82H	干衣时间 40 分钟 + 杀菌
...	...
7FH	桶自洁收费模式
...	...
CFH	干衣时间 425 分钟 + 杀菌
D0H	干衣时间 5 分钟 + 杀菌
D1H	干衣时间 10 分钟 + 杀菌
D2H	干衣时间 15 分钟 + 杀菌
D3H	干衣时间 20 分钟 + 杀菌
D4H	干衣时间 25 分钟 + 杀菌

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,    //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "",    //错误信息
  "errorinfo_cn": ""  //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) ~~async~~参数值必须为 true, nodeid 参数值必须为 1

2.27. SOS 紧急求救

http(s)://<ip>:<port>/sos

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.27.1. 获取设备的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>       //智能设备在网关设备上的节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,    //1-成功 非 1-失败
  "errorinfo": "",    //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": <true/false>, //锁当前是否是故障状态: true-故障状态 false-正常状态
    "BatteryPercentage": <percent>, //电池电量百分比: 0~100
    "CardNumber": "<card number>" // (NB 锁)NB 卡卡号
  }
}
```

【注】

- 1) NB-IoT 设备的 nodeid 参数值必须为 1

2.28. 串口设备

http(s)://<ip>:<port>/serial

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.28.1. 获取串口设备的当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>", //串口设备的逻辑节点 id
  "params": {
    "ForceRefresh": "<true/false> //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": "<true/false>", //串口设备当前是否是故障状态: true-故障状态 false-正常状态
    "Status": "<status>", //串口设备当前状态: on-开状态 off-关状态
  }
}
```

[示例]

curl -X POST \

http(s)://112.74.20.113:8888/serial \

-H 'Content-Type: application/json' \

-H 'Authorization:Bearer

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslbnRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \

-d '{

"method": "thing.service.property.get",

"logicnodeid": "iRemote2005000015000-14-0-"

}'

//返回结果:

```
{
```

```

    "resultcode": 1,
    "data": {
      "IsFault": false,
      "Status": "off"
    }
  }
}

```

2.28.2. 远程开

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.Turnon",
  "logicnodeid": "<logicnodeid>",      //串口设备的逻辑节点 id
  "params": {
    "IRData": "<hex>"                  //串口控制指令数据, HEX 格式
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",      //错误信息
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}

```

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/serial \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnon",
    "logicnodeid": "iRemote2005000015000-15-0-",
    "params": {
      "IRData": "AABBCC"
    }
  }
'

```

//返回结果:

```

{
  "resultcode": 1
}

```

2.28.3. 远程关

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.Turnoff",
  "logicnodeid": "<logicnodeid>", //串口设备的逻辑节点 id
  "params": {
    "IRData": "<hex>" //串口设备指令数据, HEX 格式
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/serial \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.Turnoff",
    "logicnodeid": "iRemote2005000015000-15-0-",
    "params": {
      "IRData": "DDEEFF"
    }
  }'
```

//返回结果:

```
{
  "resultcode": 1
}
```

2.28.4. 设置情景

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetScene",
  "logicnodeid": "<logicnodeid>",          //串口设备的逻辑节点 id
  "params": {
    "SceneID": <scene id>,                //串口设备情景 id: 1~32
    "IRData": "<hex>"                     //串口设备指令数据, HEX 格式, 最大长度 60(30 byte)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": ""       //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/serial \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.SetScene",
    "logicnodeid": "iRemote2005000015000-15-0-",
    "params": {
      "SceneID": 1,
      "IRData": "001122334455"
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.28.5. 删除情景

方法: post

请求头(header):

```
Accept-Language: <language>          //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.DeleteScene",
  "logicnodeid": "<logicnodeid>",          //串口设备的逻辑节点 id
  "params": {
    "SceneID": <scene id>                //串口设备情景 id: 1~32
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",      //错误信息
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/serial \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.DeleteScene",
    "logicnodeid": "iRemote2005000015000-15-0-",
    "params": {
      "SceneID": 1
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.28.6. 运行情景

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.RunScene",
  "logicnodeid": "<logicnodeid>", //串口设备的逻辑节点 id
  "params": {
    "SceneID": <scene id> //串口设备情景 id: 1~32
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",      //错误信息
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/serial \
```

```

-H 'Content-Type: application/json' \
-H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
-d '{
  "method": "thing.service.RunScene",
  "logicnodeid": "iRemote2005000015000-15-0-",
  "params": {
    "SceneID": 1
  }
}'
//返回结果:
{
  "resultcode": 1
}

```

2.28.7. 设置串口通讯超时时间

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.SetSerialOvertime",
  "logicnodeid": "<logicnodeid>",        //串口设备的逻辑节点 id
  "params": {
    "Overtime": <ms>                     //超时时间(毫秒)
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",      //错误信息
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}

```

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/serial \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.SetSerialOvertime",
    "logicnodeid": "iRemote2005000015000-15-0-",
    "params": {
      "Overtime": 100
    }
  }'

```

```
//返回结果:
{
  "resultcode": 1
}
```

2.28.8. 获取串口通讯超时时间

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetSerialOvertime",
  "logicnodeid": "<logicnodeid>" //串口设备的逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "params": {
    "Overtime": <ms>
  }
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/serial \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.GetSerialOvertime",
    "logicnodeid": "iRemote2005000015000-15-0-"
  }'
```

//返回结果:

```
{
  "channelid": 0,
  "deviceid": "iRemote2005000015000",
  "id": 0,
  "logicnodeid": "iRemote2005000015000-15-0-",
  "nodeid": 15,
  "params": {
    "Overtime": 10
  },
  "report": "thing.event.Overtime.post",
  "resultcode": 1,
}
```

```
"type": "notification"
}
```

2.28.9. 设置串口通讯波特率

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.SetBaudRate",
  "logicnodeid": "<logicnodeid>",        //串口设备的逻辑节点 id
  "params": {
    "BaudRate": <rate>,                  //波特率
    "CheckBit": <num>,                   //校验位: 0-无校验 1-偶校验 2-奇校验
    "StopBit": <num>                     //停止位: 0-1 位 1-1.5 位 2-2 位
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                       //1-成功 非 1-失败
  "errorinfo": "",                       //错误信息
  "errorinfo_cn": ""                     //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/serial \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.SetBaudRate",
    "logicnodeid": "iRemote2005000015000-15-0-",
    "params": {
      "BaudRate": 9600,
      "CheckBit": 0,
      "StopBit": 0
    }
  }'
```

//返回结果:

```
{
  "resultcode": 1
}
```

2.28.10. 获取串口通讯波特率

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.GetBaudRate",  
  "logicnodeid": "<logicnodeid>" //串口设备的逻辑节点 id  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "params": {  
    "BaudRate": <rate>, //波特率  
    "CheckBit": <num>, //校验位: 0-无校验 1-偶校验 2-奇校验  
    "StopBit": <num> //停止位: 0-1 位 1-1.5 位 2-2 位  
  }  
}
```

[示例]

```
curl -X POST \  
  http(s)://112.74.20.113:8888/serial \  
  -H 'Content-Type: application/json' \  
  -H 'Authorization:Bearer  
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \  
  -d '{  
    "method": "thing.service.GetBaudRate",  
    "logicnodeid": "iRemote2005000015000-15-0-"  
  }'
```

//返回结果:

```
{  
  "channelid": 0,  
  "deviceid": "iRemote2005000015000",  
  "id": 0,  
  "logicnodeid": "iRemote2005000015000-15-0-"  
  "nodeid": 15,  
  "params": {  
    "BaudRate": 9600,  
    "CheckBit": 0,  
    "StopBit": 0  
  },  
  "report": "thing.event.BaudRate.post",  
  "resultcode": 1,  
  "type": "notification"  
}
```

2.28.11. 远程控制

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.IRControl",
  "logicnodeid": "<logicnodeid>", //串口设备的逻辑节点 id
  "params": {
    "IRData": "<hex>" //串口设备指令数据, HEX 格式
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/serial \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcmVudCkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCl6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.IRControl",
    "logicnodeid": "iRemote2005000015000-15-0-",
    "params": {
      "IRData": "AABBCCDDEEFF"
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

2.29. 路灯

http(s)://**<ip>**:**<port>**/roadlamp

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.29.1. 获取路灯的当前状态

方法: post

请求头(header): Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "ForceRefresh": "<true/false> //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "IsFault": "<true/false>", //路灯当前是否是故障状态: true-故障状态 false-正常状态
    "Status": "<status>", //当前开关状态: on-开状态 off-关状态
    "Lightness": "<lightness>", //路灯的亮度程度(1~100%)
    "Voltage": "<value>", //路灯的电压
    "Current": "<value>", //路灯的电流
    "Power": "<value>", //路灯的功率
    "IMEI": "<imei>", //IMEI
    "CardNumber": "<card number>", //IMSI
    "RSSI": "<RSSI>", //RSSI
    "SNR": "<SNR>", //SNR
    "Version": "<version>" //Version
  }
}
```

2.29.2. 远程开

方法: post

请求头(header): Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.Turnon",
  "async": false, //false-同步模式(缺省), true-异步模式
}
```

```

    "logicnodeid": "<logicnodeid>" //逻辑节点 id
}
响应结果(Content-Type: application/json):
{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

【注】

当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)

2.29.3. 远程开(指定亮度百分比)

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.property.set",
    "async": false, //false-同步模式(缺省), true-异步模式
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "params": {
        "Lightness": <lightness> //路灯的亮度程度(1~100%)
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

【注】

当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)

2.29.4. 远程关

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```
{
  "method": "thing.service.Turnoff",
  "async": false,                      //false-同步模式(缺省), true-异步模式
  "logicnodeid": "<logicnodeid>"      //逻辑节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                    //1-成功 非 1-失败
  "errorinfo": "",                   //错误信息
  "errorinfo_cn": ""                 //本地化错误信息(Accept-Language 相关)
}
```

【注】

当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)

2.29.5. 设定日期定时计划

方法: post

请求头(header):

```
Accept-Language: <language>          //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.SetDateSchedule",
  "async": false,                      //false-同步模式(缺省), true-异步模式
  "logicnodeid": "<logicnodeid>",      //逻辑节点 id
  "params": {
    "ScheduleID": <number>,           //1~255
    "DateTime": "<YYYY-MM-DD HH:MM:SS>",
    "Command": {<command>}
  }
}
```

{<command>}可以是:

```
{"Action": "turnoff"}
```

或者:

```
{"Action": "turnon", "Lightness": <lightness>}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                    //1-成功 非 1-失败
  "errorinfo": "",                   //错误信息
  "errorinfo_cn": ""                 //本地化错误信息(Accept-Language 相关)
}
```

【注】

当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)

2.29.6. 删除日期定时计划

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.DeleteDateSchedule",
  "async": false, //false-同步模式(缺省), true-异步模式
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "ScheduleID": <number> //1~255
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)

2.29.7. 设定星期定时计划

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.SetWeekSchedule",
  "async": false, //false-同步模式(缺省), true-异步模式
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "ScheduleID": <number>, //1~255
    "WeekDays": <SumWeekdayValue>, //有效星期累加值: 如 5 表示星期天和星期二有效
    //1-星期天 2-星期一 4-星期二 8-星期三
    //16-星期四 32-星期五 64-星期六

    "Time": "<HH:MM:SS>",
    "Command": {<command>}
  }
}
```

{<command>}可以是:

```
{"Action": "turnoff"}
```

或者:

```
{"Action": "turnon", "Lightness": <lightness>}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",      //错误信息
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}
```

【注】

当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)

2.29.8. 删除星期定时计划

方法: post

请求头(header):

```
Accept-Language: <language>      //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.DeleteWeekSchedule",
  "async": false,                  //false-同步模式(缺省), true-异步模式
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "ScheduleID": <number>        //1~255
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",      //错误信息
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}
```

【注】

当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)

2.30. 电梯

http(s)://<ip>:<port>/lift

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.30.1. 联网授权

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.CardGrant",
  "deviceid": "<deviceid>", //电梯 DTU ID
  "nodeid": "<nodeid>", //电梯编号(2~255)
  "params": {
    "UserID": "<UserID>", //用户 ID: 1~5000
    "Serial": "<SerialNo>", //物理卡号(HEX), 人脸等虚拟卡情况不指定 Serial 值
    "ControlMode": "<mode>", //梯控模式: 0-自动选层 1-手动选层
    "FloorList": [<floorno>,<floorno>], //授权电梯楼层列表
    "ValidWeekDays": "<SumWeekdayValue>", //有效星期累加值: 如 5 表示星期天和星期二有效
    //1-星期天 2-星期一 4-星期二 8-星期三
    //16-星期四 32-星期五 64-星期六
    "ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
    "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
    "LiftInfo": { //电梯信息
      "IsABDoor":<true/false>, //是否是 AB 贯穿门
      "TotalFloors":<int>, //总楼层数
      "PublicFloorList": [<floorno>,<floorno>] //公共楼层列表
    }
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 0-用户已存在 2-锁用户数超上限 3/4-卡号重复 5-UserID 无效
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】授权楼层列表的元素 floorno 有效值 1~64, floorno 值是电梯设备本身的楼层序号(从 1 开始计数)、不是电梯所安装的楼盘的楼层号

2.30.2. 删除联网授权

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.DeleteGrant",
  "deviceid": "<deviceid>", //电梯 DTU ID
}
```

```

        "nodeid": <nodeid>, //电梯编号(2~255)
        "params": {
            "UserID": <UserID> //用户 ID: 1~5000
        }
    }
}

响应结果(Content-Type: application/json):
{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

2.30.3. 暂停授权

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.PauseGrant",
    "deviceid": "<deviceid>", //电梯 DTU ID
    "nodeid": <nodeid>, //电梯编号(2~255)
    "params": {
        "UserID": <UserID> //用户 ID: 1~5000
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

2.30.4. 恢复授权

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.RestoreGrant",
    "deviceid": "<deviceid>", //电梯 DTU ID
    "nodeid": <nodeid>, //电梯编号(2~255)
    "params": {
        "UserID": <UserID> //用户 ID: 1~5000
    }
}

```

```

    }
}
响应结果(Content-Type: application/json):
{
    "resultcode": 1,        //1-成功 非 1-失败
    "errorinfo": "",        //错误信息
    "errorinfo_cn": ""      //本地化错误信息(Accept-Language 相关)
}

```

2.30.5. 远程呼梯

方法: post

请求头(header):

```

Accept-Language: <language>        //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.RemoteControl",
    "deviceid": "<deviceid>",        //电梯 DTU ID
    "nodeid": <nodeid>,            //电梯编号(2~255)
    "params": {
        "TargetFloor": <floorno>,    //目标电梯楼层序号(1~64)
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,        //1-成功 非 1-失败
    "errorinfo": "",        //错误信息
    "errorinfo_cn": ""      //本地化错误信息(Accept-Language 相关)
}

```

【注】

- 1) floorno 值是电梯设备本身的楼层序号(从 1 开始计数)、不是电梯所在的楼盘的楼层号;
- 2) 对于 AB 门电梯: 呼 A 门时 TargetFloor 值传目标电梯楼层序号(1~32)、呼 B 门时 TargetFloor 值传目标电梯楼层序号(1~32)+总楼层数(最大 32);

2.30.6. 设置电梯信息

方法: post

请求头(header):

```

Accept-Language: <language>        //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.SetLiftInfo",
    "deviceid": "<deviceid>",        //电梯 DTU ID
    "nodeid": <nodeid>,            //电梯编号(2~255)
}

```

```

    "params": {
      "EnableManagement": <true/false>, //梯控管理功能: true-打开 false-关闭
      "IsABDoor": <true/false>, //是否是 AB 贯穿门
      "TotalFloors": <int>, //总楼层数
      "PublicFloorList": [<floorno>,<floorno>] //公共楼层列表
    }
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功 0-用户已存在 2-锁用户数超上限 3/4-卡号重复 5-UserID 无效
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

【注】授权楼层列表的元素 floorno 有效值 1~64, floorno 值是电梯设备本身的楼层序号(从 1 开始计数)、不是电梯所安装的楼盘的楼层号

2.30.7. 获取电梯信息

方法: post

请求头(header): Accept-Language: <language> //en、zh-CN
 Authorization: Bearer <JWT token>

请求参数:

```

{
  "method": "thing.service.GetLiftInfo",
  "deviceid": "<deviceid>", //电梯 DTU ID
  "nodeid": <nodeid> //电梯编号(2~255)
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "params": {
    "EnableManagement": <true/false>, //梯控管理功能: true-打开 false-关闭
    "IsABDoor": <true/false>, //是否是 AB 贯穿门
    "TotalFloors": <int>, //总楼层数
    "PublicFloorList": [<floorno>,<floorno>] //公共楼层列表
  }
}

```

2.30.8. 获取蓝牙乘梯参数

方法: post

请求头(header): Accept-Language: <language> //en、zh-CN
 Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetBluetoothLift",
  "deviceid": "<deviceid>",          //电梯 DTU ID
  "nodeid":  <nodeid>                //电梯编号(2~255)
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
  "params": {
    "Status": "<on/off>",      //状态: on-打开 off-关闭
    "Sensitivity": <number>    //灵敏度(40~110)
  }
}
```

2.30.9. 设置蓝牙乘梯参数

方法: post

请求头(header): Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetBluetoothLift",
  "deviceid": "<deviceid>",          //电梯 DTU ID
  "nodeid":  <nodeid>,              //电梯编号(2~255)
  "params": {
    "Status": "<on/off>",      //状态: on-打开 off-关闭
    "Sensitivity": <number>    //灵敏度: 40~110, 或 0(缺省灵敏度)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": ""        //本地化错误信息(Accept-Language 相关)
}
```

2.30.10. 设置蓝牙乘梯灵敏度

方法: post

请求头(header): Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetBluetoothLiftSensitivity",
  "deviceid": "<deviceid>",          //电梯 DTU ID
  "nodeid": "<nodeid>",              //电梯编号(2~255)
  "params": {
    "Sensitivity":<number>          //灵敏度: 40~110, 或 0(缺省灵敏度)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": ""       //本地化错误信息(Accept-Language 相关)
}
```

2.31. 电梯对讲设备

http(s)://<ip>:<port>/sip

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.31.1. 读取配置

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetConfig",
  "deviceid": "<deviceid>",          //对讲设备 ID
  "nodeid": 1,
  "params": { //可指定下面 1 至多个参数项
    "call": 1,
    "location": 1,
    "sip": 1,
    "vol": 1,
    "net": 1,
    "card": 1,
    "mver": 1,
    "port": 1,
    "cpu": 1,
    "sms": 1,
    "tl": 1,
    "sm": 1,
  }
}
```

```

        "ctime": 1
    }
}

```

【注】params 参数项说明:

参数项	类型	说明
call	int	需要获取呼叫管理机的地址 包含 6 个地址, =1 表示读取这个字段
location	int	分机的安装位置, =1 表示读取这个字段
sip	int	注册到 SIP 服务器 用户名 服务器地址 密码, =1 表示读取这个字段
vol	int	录音音量和播放音量, =1 表示读取这个字段
net	int	当前的分机地址 IP 子网掩码 网关地址, =1 表示读取这个字段
card	int	物联卡的一些消息消息 ICCID IMSI ,=1 表示读取这个字段
mver	int	4G 模块的版本号, =1 表示读取这个字段
port	int	当前分机支持端口,=1 表示读取这个字段
cpu	int	cpu 编号, =1 表示读取这个字段
sms	int	iS101 语音分机设使用打电话的时候是否发送短信= 1 表示读取这个字段
tl	int	iM102 语音网关主叫电话读取, = 1 表示读取这个字段
sm	int	iM102 语音网关是否发送短信, = 1 表示读取这个字段
ctime	int	通话时间, = 1 表示读取这个字段

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //成功
  "data": {
    "name": "对讲主机 1",
    "cl_1": "3B439900096193CE",
    "cl_2": "3B439900096193CE",
    "cl_3": "3B439900096193CE",
    "cl_4": "3B439900096193CE",
    "cl_5": "3B439900096193CE",
    "cl_6": "3B439900096193CE",
    "location": "1 栋 1 单元 2 号电梯",
    "sip_se": "sip.isurpss.com.cn",
    "sip_port": 15064,
    "vol_h_mic": 30,
    "vol_h_spk": 70,
    "vol_g_mic": 20,
    "vol_g_spk": 60,
    "vol_m_mic": 20,
    "vol_m_spk": 90,
    "vol_ring": 60,
    "net_ip": "192.168.78.10",
    "net_mask": "255.255.255.0",
    "net_gwy": "192.168.78.1",
    "card_iccid": "89860...",
    "card_imsi": "169xxxx",
    "m_rssi": 25,
    "mver": "MC800_MD4_2C05PS_F2_28003",
    "imei": "14564664",
    "cpu": "W800",
    "sms": 1,
    "port": 4,
    "tl_1": "18925854689",
    "tl_2": "NULL",
  }
}

```

```

        "tl_3": "NULL",
        "tl_4": "NULL",
        "tl_5": "NULL",
        "sm_1": 1,
        "sm_2": 0,
        "sm_3": 0,
        "sm_4": 0,
        "sm_5": 0,
        "ctime": 2
    }
}

```

【注】data 数据项说明:

数据项	类型	说明
type	string	设备类型
name	string	设备名称
cl_1	string	分机呼叫地址 1(只有分机才有的字段)
cl_2	string	分机呼叫地址 2(只有分机才有的字段)
cl_3	string	分机呼叫地址 3(只有分机才有的字段)
cl_4	string	分机呼叫地址 4(只有分机才有的字段)
cl_5	string	分机呼叫地址 5(只有分机才有的字段)
cl_6	string	分机呼叫地址 6(只有分机才有的字段)
location	string	设备的安装位置描述消息
sip_se	string	SIP 服务器地址
sip_port	int	Sip 服务器注册的端口号
vol_h_mic	int	分机的录音音量(只有分机才有的字段)20-90
vol_h_spk	int	分机的播放音量(只有分机才有的字段)20-90
vol_g_mic	int	管理机的手柄录音音量(管理机有的字段)20-90
vol_g_spk	int	管理机的手柄播放音量(管理机有的字段)20-90
vol_m_mic	int	管理机的免提录音音量(管理机有的字段)20-90
vol_m_spk	int	管理机的免提播放音量(管理机有的字段)20-90
vol_ring	int	铃声大小(管理机有的字段)20-90
net_ip	string	设备的 IP 地址(4G 设备无效)
net_mask	string	设备的子网掩码(4G 设备无效)
net_gwy	string	设备的网关地址(4G 设备无效)
card_iccid	string	物联网卡的 iccid (4G 设备)
card_imsi	string	物联网卡的移动串号 (4G 设备)
m_rssi	string	4G 模块的信号强度 (4G 设备)
mver	string	4G 模块的版本号 (4G 设备)
imei	string	4G 模块的串号 (4G 设备)
cpu	string	CPU 版本号
sms	int	iS101 语音分机, 是否开始接受短信命令 1 开启 0 不开
port	int	当前设备局数,只有分机才有
tl_1	string	语音网关主呼叫号码 1 1345854689 默认是 NULL 注: 只有语音网关 iM102 才有的字段
tl_2	string	语音网关主呼叫号码 2 默认是 NULL 注: 只有语音网关 iM102 才有的字段
tl_3	string	语音网关主呼叫号码 3 默认是 NULL 注: 只有语音网关 iM102 才有的字段
tl_4	string	语音网关主呼叫号码 4 默认是 NULL

		注：只有语音网关 iM102 才有的字段
tl_5	string	语音网关主呼叫号码 5 默认是 NULL 注：只有语音网关 iM102 才有的字段
sm_1	int	语音网关主呼叫号码 1 是否接受短信 默认是 0 值 0-1 注：只有语音网关 iM102 才有的字段
sm_2	int	语音网关主呼叫号码 2 是否接受短信 默认是 0 值 0-1 注：只有语音网关 iM102 才有的字段
sm_3	int	语音网关主呼叫号码 3 是否接受短信 默认是 0 值 0-1 注：只有语音网关 iM102 才有的字段
sm_4	int	语音网关主呼叫号码 4 是否接受短信 默认是 0 值 0-1 注：只有语音网关 iM102 才有的字段
sm_5	int	语音网关主呼叫号码 5 是否接受短信 默认是 0 值 0-1 注：只有语音网关 iM102 才有的字段
ctime	int	通话时间 可选参数 1-5 单位分钟 出厂默认 2 分钟

2.31.2. 下发配置

方法：post

请求头(header)：

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数：

```
{
  "method": "thing.service.SetConfig",
  "deviceid": "<deviceid>", //对讲设备 ID
  "nodeid": 1,
  "params": { //可指定下面 1 至多个参数项
    "name": "xxx",
    "cl_1": "isipxxx",
    "cl_2": "isipxxx",
    "cl_3": "isipxxx",
    "cl_4": "isipxxx",
    "cl_5": "isipxxx",
    "cl_6": "isipxxx",
    "location": "1 栋 1 单元 2 号电梯",
    "sip_se": "sip.isurpass.com.cn",
    "sip_port": 5060,
    "vol_h_mic": 30,
    "vol_h_spk": 70,
    "vol_g_mic": 20,
    "vol_g_spk": 60,
    "vol_m_mic": 20,
    "vol_m_spk": 90,
    "vol_ring": 60,
    "net_ip": "192.168.78.10",
    "net_mask": "255.255.255.0",
    "net_gwy": "192.168.78.1",
    "sms": 1,
  }
}
```

```

    "ctime": 2
}
}

```

【注】params 参数项说明:

参数项	类型	说明
name	string	设备名称
cl_1	string	分机呼叫地址 1(只有分机才有的字段)
cl_2	string	分机呼叫地址 2(只有分机才有的字段)
cl_3	string	分机呼叫地址 3(只有分机才有的字段)
cl_4	string	分机呼叫地址 4(只有分机才有的字段)
cl_5	string	分机呼叫地址 5(只有分机才有的字段)
cl_6	string	分机呼叫地址 6(只有分机才有的字段)
location	string	设备的安装位置描述消息
sip_se	string	SIP 服务器地址
sip_port	int	Sip 服务器注册的端口号
vol_h_mic	int	分机的录音音量(只有分机才有的字段)20-90
vol_h_spk	int	分机的播放音量(只有分机才有的字段)20-90
vol_g_mic	int	管理机的手柄录音音量(管理机有的字段)20-90
vol_g_spk	int	管理机的手柄播放音量(管理机有的字段)20-90
vol_m_mic	int	管理机的免提录音音量(管理机有的字段)20-90
vol_m_spk	int	管理机的免提播放音量(管理机有的字段)20-90
vol_ring	int	铃声大小(管理机有的字段)20-90
net_ip	string	设备的 IP 地址(4G 设备无效)
net_mask	string	设备的子网掩码(4G 设备无效)
net_gwy	string	设备的网关地址(4G 设备无效)
sms	int	iS101 才有的数据 是否开始接受短信命令 1 开启 0 不开
tl_1	string	语音网关主呼叫号码 1 1345854689 默认是 NULL 注: 只有语音网关 iM102 才有的字段
tl_2	string	语音网关主呼叫号码 2 默认是 NULL 注: 只有语音网关 iM102 才有的字段
tl_3	string	语音网关主呼叫号码 3 默认是 NULL 注: 只有语音网关 iM102 才有的字段
tl_4	string	语音网关主呼叫号码 4 默认是 NULL 注: 只有语音网关 iM102 才有的字段
tl_5	string	语音网关主呼叫号码 5 默认是 NULL 注: 只有语音网关 iM102 才有的字段
sm_1	int	语音网关主呼叫号码 1 是否接受短信 默认是 0 值 0-1 注: 只有语音网关 iM102 才有的字段
sm_2	int	语音网关主呼叫号码 2 是否接受短信 默认是 0 值 0-1 注: 只有语音网关 iM102 才有的字段
sm_3	int	语音网关主呼叫号码 3 是否接受短信 默认是 0 值 0-1 注: 只有语音网关 iM102 才有的字段
sm_4	int	语音网关主呼叫号码 4 是否接受短信 默认是 0 值 0-1 注: 只有语音网关 iM102 才有的字段
sm_5	int	语音网关主呼叫号码 5 是否接受短信 默认是 0 值 0-1 注: 只有语音网关 iM102 才有的字段
ctime	int	通话时间 可选参数 1-5 单位分钟 出厂默认 2 分钟

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1  //1: 成功
}
```

2.31.3. 读取电话本

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.GetAddrBook",
  "deviceid": "<deviceid>",           //对讲设备 ID
  "nodeid": 1
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,  //成功
  "data": {
    "addr": [
      {
        "device_ip": "isipxxx",  //分机 ID
        "Elevator": [
          {
            "PhoneNumber": "00000999",
            "QuickKeyName": "一栋一单元一号梯",
            "Interface": 1
          }
        ]
      }
    ],
    ...
  ]
}
```

【注】data 数据项说明:

字段名	数据类型	说明
addr	Obj	表示是一个地址表 Array
device_ip	string	局域网管理机是 IP 4G 主机是分机的 SN
Elevator	Obj	包含一个端口的详细数据 一个分机多端口的就多个具体数据最多包含 4 个端口 (Array)
PhoneNumber	string	拨号代码 8 位对齐不足 8 位前面填 0 用来管理机拨号分机使用的代码 一般编码格式 XX XX XX XX 小区号 栋号 单元号 电梯号
QuickKeyName	string	电梯的名字 例: 一栋一单元一号梯

Interface	int	分机的局端口编号 1 到 4
-----------	-----	----------------

2.31.4. 下发电话本

方法：post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetAddrBook",
  "deviceid": "<deviceid>", //对讲设备 ID
  "nodeid": 1,
  "params": {
    "addr": [
      {
        "device_ip": "<isipxxx>", //分机 ID
        "Elevator": [
          {
            "PhoneNumber": "00000999",
            "QuickKeyName": "一栋一单元一号梯",
            "Interface": 1
          }
        ]
      }
    ],
    ...
  }
}
```

【注】params 参数项说明:

字段名	数据类型	说明
addr	Obj	表示是一个地址表 Array
device_ip	string	局域网管理机是 IP 4G 主机是分机的 SN
Elevator	Obj	包含一个端口的详细数据 一个分机多端口的就多个具体数据最多包含 4 个端口 (Array)
PhoneNumber	string	拨号代码 8 位对齐不足 8 位前面填 0 用来管理机拨号分机使用的代码 一般编码格式 XX XX XX XX 小区号 栋号 单元号 电梯号
QuickKeyName	string	电梯的名字 例： 一栋一单元一号梯
Interface	int	分机的局端口编号 1 到 4

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1 //1: 成功
}
```

2.31.5. 远程重启

方法：post

请求头(header)：

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数：

```
{
  "method": "thing.service.Reboot",
  "deviceid": "<deviceid>", //对讲设备 ID
  "nodeid": 1,
  "params": {
    "sec": 0 //延时 N 秒后重启，0 立即重启
  }
}
```

响应结果(Content-Type: application/json)：

```
{
  "resultcode": 1 //1: 成功
}
```

2.31.6. 设备升级

方法：post

请求头(header)：

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数：

```
{
  "method": "thing.service.Upgrade",
  "deviceid": "<deviceid>", //对讲设备 ID
  "nodeid": 1,
  "params": {
    "file": "http://dev.isurpass.com.cn/xxx",
    "md5": "F6CFFAF6E457C00D6A543290760F360C",
    "user": "admin",
    "passwd": "12345678"
  }
}
```

【注】params 参数项说明：

参数项	数据类型	说明
file	string	文件的 HTTP OTA 下载的地址
md5	string	ota 文件 MD5 校验码
user	string	如果需要用户名
passwd	string	如果需要密码

响应结果(Content-Type: application/json)：

```
{
  "resultcode": 1  //1: 成功
}
```

2.32. 断路器

http(s)://<ip>:<port>/device

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

2.32.1. 获取当前状态

方法: post

ForceRefresh

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.property.get",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //断路器设备在网关设备上的节点 id
  "params": {
    "ForceRefresh": "<true/false> //false(缺省): 直接返回缓存, true: 访问设备&刷新缓存后返回
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "AVCP": { //A 厢电压电流功率
      "C": { //电流
        "Scale": "A",
        "Value": "<value>"
      },
      "P": { //功率
        "Scale": "W",
        "Value": "<value>"
      },
      "V": { //电压
        "Scale": "V",
        "Value": "<value>"
      }
    },
    "BVCP": { //B 厢电压电流功率
```

```

    "C": {                                //电流
      "Scale": "A",
      "Value": "<value>"
    },
    "P": {                                //功率
      "Scale": "W",
      "Value": "<value>"
    },
    "V": {                                //电压
      "Scale": "V",
      "Value": "<value>"
    }
  },
  "CVCP": {                              //C 厢电压电流功率
    "C": {                                //电流
      "Scale": "A",
      "Value": "<value>"
    },
    "P": {                                //功率
      "Scale": "W",
      "Value": "<value>"
    },
    "V": {                                //电压
      "Scale": "V",
      "Value": "<value>"
    }
  },
  "IsFault": "<false/true>",             //是否故障
  "Kwh": "<value>",                       //当前能耗
  "Status": "<on/off>",                  //当前状态: on-合闸 off-分闸
  "Temperature": {                       //当前温度
    "Scale": "C",
    "Value": "<value>"
  }
}
}

```

2.32.2. 远程合闸

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.Turnon",
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid": "<nodeid>"                  //断路器设备在网关设备上的节点 id
}

```

响应结果(Content-Type: application/json):

```

{

```

```

    "resultcode": 1,      //1-成功 非 1-失败
    "errorinfo": "",     //错误信息
    "errorinfo_cn": ""   //本地化错误信息(Accept-Language 相关)
}

```

2.32.3. 远程分闸

方法: post

请求头(header):

```

Accept-Language: <language>      //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.Turnoff",
  "deviceid": "<deviceid>",        //网关设备 ID
  "nodeid": <nodeid>              //断路器设备在网关设备上的节点 id
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",     //错误信息
  "errorinfo_cn": ""   //本地化错误信息(Accept-Language 相关)
}

```

2.32.4. 设置定时断合

方法: post

请求头(header):

```

Accept-Language: <language>      //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.SetTimer",
  "deviceid": "<deviceid>",        //网关设备 ID
  "nodeid": <nodeid>,              //断路器设备在网关设备上的节点 id
  "params": {
    "ID": <id>,                     //id: 1~20
    "Time": "<HH:MM>",
    "WeekDays": <SumWeekdayValue>, //星期累加值: 如 5 表示星期天和星期二有效
                                     1-星期天 2-星期一 4-星期二 8-星期三
                                     16-星期四 32-星期五 64-星期六
    "Action": "<switchon/switchoff>" //switchon-合闸 switchoff-断开
  }
}

```

响应结果(Content-Type: application/json):

```

{

```

```
"resultcode": 1,      //1-成功 非 1-失败
"errorinfo": "",      //错误信息
"errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}
```

2.32.5. 删除定时断合

方法: post

请求头(header):

```
Accept-Language: <language>      //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.DelTimer",
  "deviceid": "<deviceid>",        //网关设备 ID
  "nodeid": <nodeid>,             //断路器设备在网关设备上的节点 id
  "params": {
    "ID": <id>                     //id: 1~20
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",      //错误信息
  "errorinfo_cn": ""    //本地化错误信息(Accept-Language 相关)
}
```

2.32.6. 设置上报周期

方法: post

请求头(header):

```
Accept-Language: <language>      //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.SetPeroid",
  "deviceid": "<deviceid>",        //网关设备 ID
  "nodeid": <nodeid>,             //断路器设备在网关设备上的节点 id
  "params": {
    "Minute": <value>
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",      //错误信息
}
```

```
"errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

2.32.7. 设置告警阈值

方法: post

请求头(header):

```
Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.SetWarningThreshold",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //断路器设备在网关设备上的节点 id
  "params": {
    "Overload": { //过载
      "Enable": "<true/false>", //true-开启 false-关闭
      "Threshold": "<value>" //阈值(1~13Kws)
    },
    "Leakage": { //漏电
      "Enable": "<true/false>", //true-开启 false-关闭
      "Threshold": "<value>" //阈值(10~90mA)
    },
    "Temperature": { //高温
      "Enable": "<true/false>", //true-开启 false-关闭
      "Threshold": "<value>" //阈值(40~150 度)
    },
    "Overcurrent": { //过流
      "Enable": "<true/false>", //true-开启 false-关闭
      "Threshold": "<value>" //阈值(1~100A)
    },
    "Overvoltage": { //过压
      "Enable": "<true/false>", //true-开启 false-关闭
      "Threshold": "<value>" //阈值(250~300V)
    },
    "Undervoltage": { //欠压
      "Enable": "<true/false>", //true-开启 false-关闭
      "Threshold": "<value>" //阈值(150~200V)
    }
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

3. WebSocket 服务

3.1. 认证

JWT 认证方式， 详见 [JWT Token 定义](#)

*建立连接后发送 Bearer <JWT token>,<Accept-Language>

3.2. 请求 z-wave 网关设备进入/退出添加&删除设备状态

url: ws://<ip>:<port>/ws/addDelDevice

encoding: UTF-8

*生产环境使用 wss

1) 请求网关设备进入添加&删除设备状态:

{"method":"thing.service.EnterAddDelDeviceStatus", "deviceid":"<deviceid>"}

2) 请求网关设备退出添加&删除设备状态:

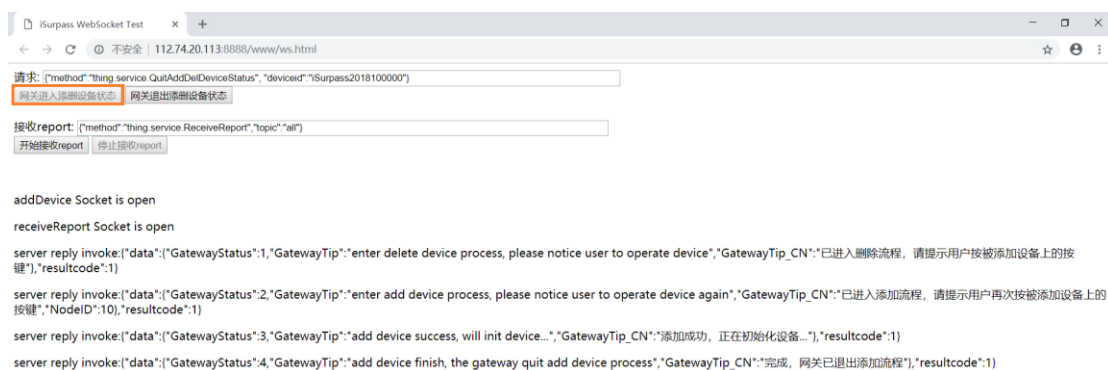
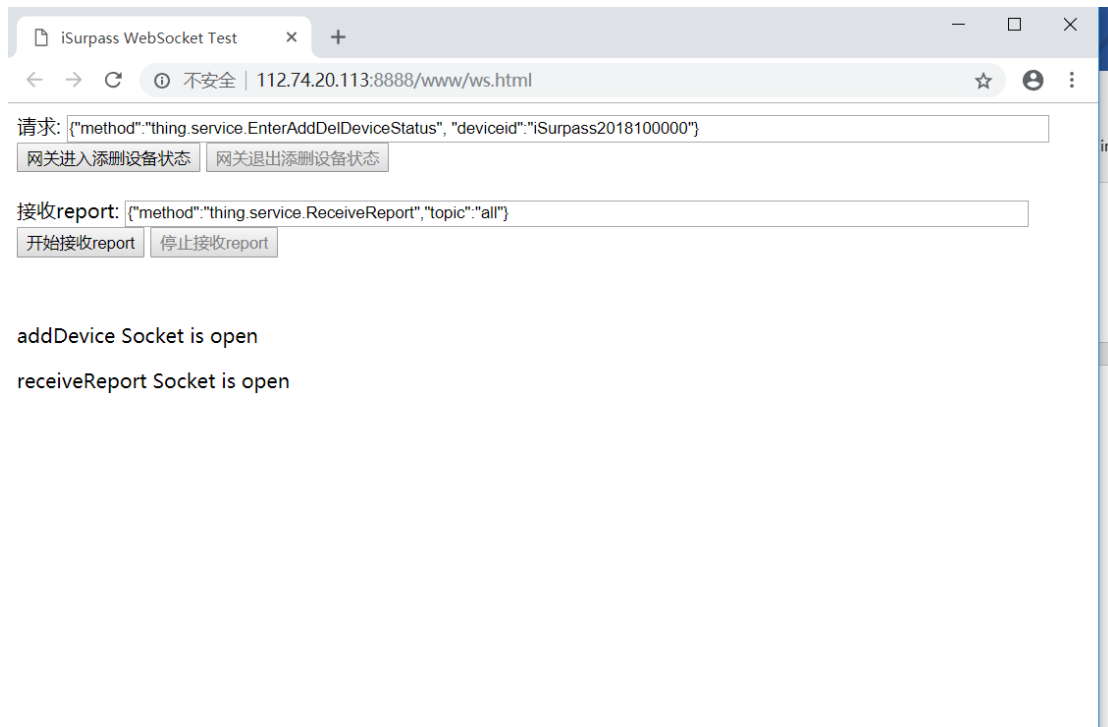
{"method":"thing.service.QuitAddDelDeviceStatus", "deviceid":"<deviceid>"}

【注】

1) 该接口仅适用于 z-wave 网关

示例:

[http\(s\)://112.74.20.113:8888/www/ws.html](http(s)://112.74.20.113:8888/www/ws.html)



3.3. 接收设备的 report

url: ws://<ip>:<port>/ws/receiveReport

encoding: UTF-8

*生产环境使用 wss

3.3.1. 请求接收指定 topic 的 report

3.3.1.1. 请求接收全部设备 report

```
{"method":"thing.service.ReceiveReport","topic":"all"}
```

3.3.1.2. 请求接收指定网关设备的 report

```
{"method":"thing.service.ReceiveReport", "topic":"<deviceid>"}
```

3.3.1.3. 请求接收指定网关下的指定节点设备的 report

```
{"method":"thing.service.ReceiveReport", "topic":"<deviceid>/<nodeid>"}
```

【注】 report 内容详见 [设备的 report](#)

3.3.2. 请求停止接收指定 topic 的 report

3.3.2.1. 请求停止接收全部设备 report

```
{"method":"thing.service.UnReceiveReport","topic":"all"}
```

3.3.2.2. 请求停止接收指定网关下的 report

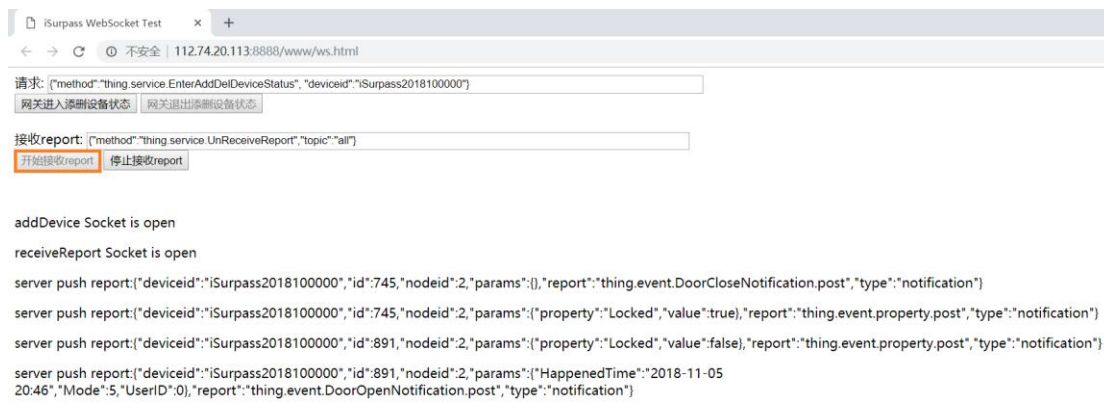
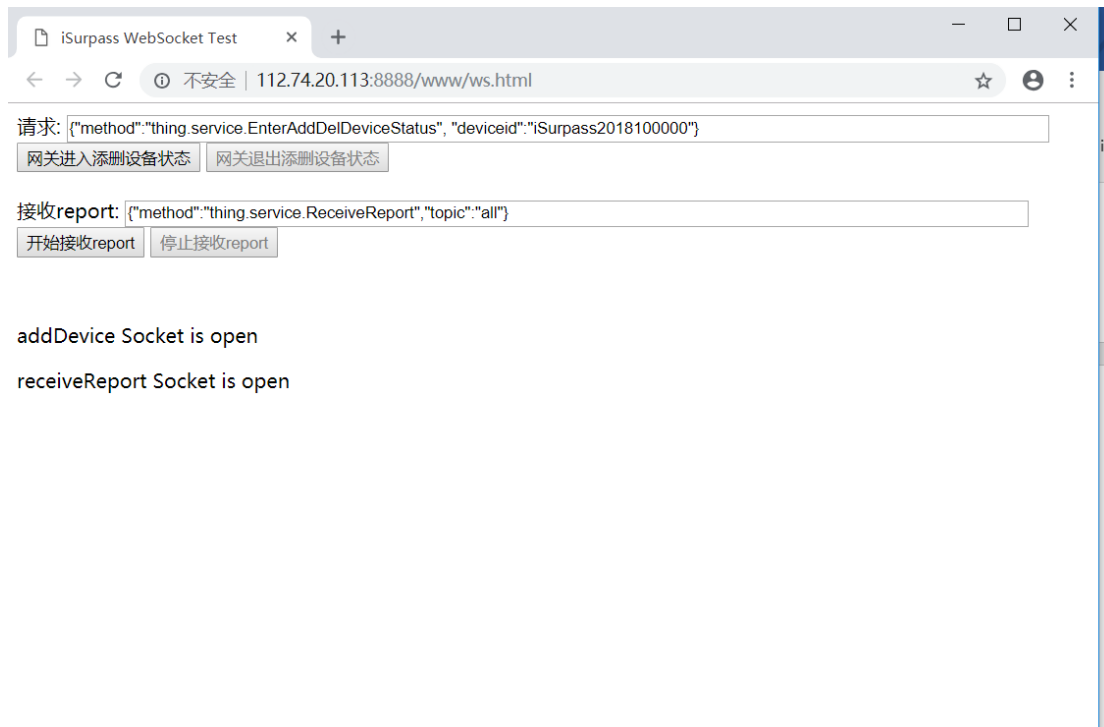
```
{"method":"thing.service.UnReceiveReport ", "topic":"<deviceid>"}
```

3.3.2.3. 请求停止接收指定网关下的指定节点设备的 report

```
{"method":"thing.service.UnReceiveReport ", "topic":"<deviceid>/<nodeid>"}
```

示例：

[http\(s\)://112.74.20.113:8888/www/ws.html](http(s)://112.74.20.113:8888/www/ws.html)



3.4. 设备的 report

3.4.1. 网关设备的 report

3.4.1.1. 增加节点设备 report

```
{
  "id": <id>,           //report id
  "deviceid": "<deviceid>", //网关设备 id
  "report": "thing.event.NewNode.post",
```

```

"type": "notification",
"params": {
  "NodeID": "<NodeID>", //新节点的 nodeid
  "DeviceType": "lock", //新节点的设备类型
  "DeviceType_CN": "lock", //设备类型名称, 与 Accept-Language 相关
  "DeviceModel": "<DeviceModel>", //新节点的设备厂商型号 ID
  "ProtocolType": "lock",
  "DSK": "", //设备的 DSK
},
"ts": <timestamp>, //Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.1.2. 删除节点设备 report

```

{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "report": "thing.event.DeleteNode.post",
  "type": "notification",
  "params": {
    "NodeID": "<NodeID>" //已删除节点的 nodeid
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.1.3. 网关设备上线 report

```

{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "report": "thing.event.Online",
  "type": "notification",
  "params": {
    "IP": "<IP>", //网关设备 IP
    "DeviceType": "<device type>", //网关设备类型:
    gateway: Z-Wave 网关/TCP 网关
    fingerprintreader: 指纹采集器
    "ProtocolType": "<protocol type>", //转换协议类型
    "DeviceModel": "<device model>", //设备模型
    "SupportDSK": <true/false> //是否支持 DSK 设备注册
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

```
}
```

3.4.1.4. 网关设备离线 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "report": "thing.event.Offline",
  "type": "notification",
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.1.5. (wifi/蜂窝移动)信号强度 report

```
{
  "deviceid": "<deviceid>",
  "report": "thing.event.property.post",
  "type": "notification",
  "params": {
    "property": "RSSI",
    "property_cn": "RSSI", //本地化属性名称, 与 Accept-Language 相关
    "value": <value> //信号强度值
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.1.6. 节点故障 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "report": "thing.event.NodeFault.post",
  "type": "notification",
  "params": {
    "NodeID": <NodeID> //故障节点的 nodeid
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.1.7. 网关设备复位 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "report": "thing.event.Reset.post",
  "type": "notification",
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.1.8. 网关扩展属性变更 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "report": "thing.event.DeviceExtendedAttribute.post",
  "type": "notification",
  "params": {
    "Name": "<value>" //扩展属性名: 扩展属性值
    ...
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.1.9. 节点扩展属性变更 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //节点设备 id
  "report": "thing.event.NodeExtendedAttribute.post",
  "type": "notification",
  "params": {
    "Name": "<value>" //扩展属性名: 扩展属性值
    ...
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.1.10. 逻辑节点扩展属性变更 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //节点设备 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.LogicNodeExtendedAttribute.post",
  "type": "notification",
  "params": {
    "Name": "<value>" //扩展属性名: 扩展属性值
    ...
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.1.11. 增加无源设备 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "report": "thing.event.AddNaturalDevice.post",
  "type": "notification",
  "params": {
    "NodeID": <NodeID>, //新节点的 nodeid
    "DeviceType": "natural", //新节点的设备类型
    "DeviceType_CN": "natural", //设备类型名称, 与 Accept-Language 相关
    "DeviceModel": "<DeviceModel>", //新节点的设备厂商型号 ID
    "ProtocolType": "natural"
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.1.12. (TCP 网关)节点连接 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "report": "thing.event.NodeConnected.post",
  "type": "notification",
  "params": {
    "NodeID": <NodeID>, //节点的 nodeid
    "ConnectionType": "<tcp/websocket>", //连接类型
    "IP": "<ip>", //IP
  }
}
```

```

        "Port": <int>                                //Port
    },
    "ts": <timestamp>,                                //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>"                     //日期时间
}

```

3.4.1.13. (TCP 网关)节点断开连接 report

```

{
    "id": <id>,                                //report id
    "deviceid": "<deviceid>", //网关设备 id
    "report": "thing.event.NodeDisconnected.post",
    "type": "notification",
    "params": {
        "NodeID": <NodeID>,                    //节点的 nodeid
        "ConnectionType": "<tcp/websocket>"     //连接类型
    },
    "ts": <timestamp>,                                //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>"                 //日期时间
}

```

3.4.2. 锁的 report

3.4.2.1. 锁最新状态 report

```

{
    "id":<id>,                                //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>,                        //锁设备在网关设备上的节点 id
    "report": "thing.event.property.post",
    "type": "notification", //notification-通知
    "params": {
        "property": "Locked",
        "property_cn": "lock status",          //本地化属性名称, 与 Accept-Language 相关
        "value": <true/false>,                 //锁当前是否是落锁状态: true-落锁状态 false-开锁状态
        "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
        "HappenedZone": <time zone>,           //设备的时区
        "Happened_ts": <happened timestamp>     //设备事件发生 Unix 时间戳(秒)
    },
    "ts": <timestamp>,                                //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>",                 //系统处理事件的日期时间
    "report_cn": "<report event name>",             //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>"       //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.2. 锁无线信号 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,   //锁设备在网关设备上的节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "RSSI",
    "property_cn": "RSSI", //本地化属性名称, 与 Accept-Language 相关
    "value":<percent>,    //信号强度百分比: 0~100
  },
  "ts": <timestamp>,      //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.2.3. 锁电池电量 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,   //锁设备在网关设备上的节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "BatteryPercentage",
    "property_cn": "battery percentage ", //本地化属性名称, 与 Accept-Language 相关
    "value":<percent>,                  //电池电量百分比: 0~100
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": <time zone>,         //设备的时区
    "Happened_ts": <happened timestamp>  //设备事件发生 Unix 时间戳(秒)
  },
  "ts": <timestamp>,      //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.2.4. 开门通知 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,   //锁设备在网关设备上的节点 id
  "report":"thing.event.DoorOpenNotification.post",
}
```

```

"type":"notification",          //notification-通知
"params":{
  "Mode": "<mode>",           //1-密码 2-指纹 3-卡 4-身份证 5-App 远程 7-人脸 8-室内 9-蓝牙
                                10-动态密码 90-蓝牙钥匙扣 91-二维码
                                21-密码指纹 22-密码卡片 23-指纹卡片 24-双密码 25-双指纹 26-双卡片
  "Mode_CN": "<mode name>",    //开锁模式名称, 与 Accept-Language 相关
  "UserID": "<userid>",         //锁用户 id
  "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
  "HappenedZone": "<time zone>",      //设备的时区
  "Happened_ts": "<happened timestamp>", //设备事件发生 Unix 时间戳(秒)
  "UserName": "<username>",         //锁用户名
  "Password": "<password>",         //((仅当 Mode 为 10 时)开锁密码
  "UserIDs": [<userid>, <userid>], //only for 组合开锁(Mode 21 ~ 26)
  "UserNames": [<username>,<username>] //only for 组合开锁(Mode 21 ~ 26)
},
"ts": <timestamp>,              //系统处理事件的 Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>",    //系统处理事件的日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.5. 关门通知 report

```

{
  "id":<id>,                    //report id
  "deviceid": "<deviceid>",      //网关设备 id
  "nodeid":<nodeid>,           //锁设备在网关设备上的节点 id
  "report":"thing.event.DoorCloseNotification.post",
  "type":"notification",
  "params":{
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": "<time zone>",         //设备的时区
    "Happened_ts": <happened timestamp>    //设备事件发生 Unix 时间戳(秒)
  },
  "ts": <timestamp>,            //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>",        //系统处理事件的日期时间
  "report_cn": "<report event name>",     //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.6. 锁电池低电量告警 report

```

{
  "id":<id>,                    //report id
  "deviceid": "<deviceid>",      //网关设备 id
  "nodeid":<nodeid>,           //锁设备在网关设备上的节点 id
  "report":"thing.event.ElectricityAlarm.post",
  "type":"alarm",              //alarm-告警
  "params":{
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)

```

```

        "HappenedZone": <time zone>, //设备的时区
        "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.7. 取消反锁 report

```

{
    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
    "report": "thing.event.DoorUnlockedFromInside.post",
    "type": "notification",
    "params": {
        "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
        "HappenedZone": <time zone>, //设备的时区
        "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.8. 锁设备告警 report

```

{
    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
    "report": "thing.event.DeviceAlarm.post",
    "type": "alarm", //alarm-告警
    "params": {
        "AlarmType": <AlarmType>, //告警类型: 1-防撬 3-试错 5 次以上 9-劫持 11-门未锁上
        "TipInfo": "<TipInfo>", //告警提示信息
        "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
        "HappenedZone": <time zone>, //设备的时区
        "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.9. 反锁 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
  "report": "thing.event.DoorLockedFromInside.post",
  "type": "notification",
  "params": {
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": <time zone>, //设备的时区
    "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.2.10. 添加/修改锁用户 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
  "report": "thing.event.AddUser.post",
  "type": "lockuser",
  "params": {
    "UserType": <UserType>, //类型: 1-密码 2-指纹 3-MF 卡 4-身份证 7-人脸
    // 9-蓝牙/二维码 90-蓝牙钥匙扣
    "UserType_CN": "<user type name>", //类型名称, 与 Accept-Language 相关
    "UserLevel": <UserLevel>, //级别: 1-管理员 2-普通用户 4-临时密码(UserType 必须为 1)
    "UserLevel_CN": "<user level name>", //级别名称, 与 Accept-Language 相关
    "UserID": <UserID>, //UserID 规则:
    // 管理员: 1~9
    // 普通用户: 10~230
    // 一次性临时密码: 243~252
    "UserPassword": "<UserPassword>", //密码串或卡号
    // 卡数据: MF 卡, 4byte 值转换为十六进制串
    // 身份证卡, 8byte 值转换为十六进制串
    "ValidWeekDays": <SumWeekdayValue>, //有效星期累加值: 如 5 表示星期天和星期二有效
    // 1-星期天 2-星期一 4-星期二 8-星期三
    // 16-星期四 32-星期五 64-星期六
    "ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
    "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
    "UserName": "<lock user name>",
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": <time zone>, //设备的时区
    "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
}
```

```

    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.11. 修改锁用户有效日期时间 report

```

{
    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
    "report": "thing.event.UpdateUserValidDateTime.post",
    "type": "lockuser",
    "params": {
        "UserID": <UserID>, //锁用户 ID
        "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
        "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
        "UserName": "<lock user name>",
        "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
        "HappenedZone": <time zone>, //设备的时区
        "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.12. 修改锁用户有效星期时间 report

```

{
    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
    "report": "thing.event.UpdateUserValidWeekTime.post",
    "type": "lockuser",
    "params": {
        "UserID": <UserID>, //锁用户 ID
        "ValidWeekDays": <SumWeekdayValue>, //有效星期累加值: 如 5 表示星期天和星期二有效
        //1-星期天 2-星期一 4-星期二 8-星期三
        //16-星期四 32-星期五 64-星期六
        "ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间
        "UserName": "<lock user name>",
        "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
        "HappenedZone": <time zone>, //设备的时区
        "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

```
}
```

3.4.2.13. 批量修改锁用户有效日期时间 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": "<nodeid>", //锁设备在网关设备上的节点 id
  "report": "thing.event.BatchUpdateUserValidDateTime.post",
  "type": "lockuser",
  "params": {
    "UserIDs": [ "<UserID>" ], //锁用户(1 至多个)
    "UserType": <UserType>, //0-全部 1-密码 2-指纹 3-MF 卡 4-身份证
    "UserType_CN": "<user type name>", //类型名称, 与 Accept-Language 相关
    "UserLevel": <UserLevel>, //0-全部 1-管理员 2-普通用户 4-一次性临时密码
    "UserLevel_CN": "<user level name>", //级别名称, 与 Accept-Language 相关
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
    "ValidEndDateTime": "<YYYY-MM-DD HH:MM>" //有效期的终止日期时间
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.2.14. 删除锁用户 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": "<nodeid>", //锁设备在网关设备上的节点 id
  "report": "thing.event.DeleteUser.post",
  "type": "lockuser",
  "params": {
    "UserType": <UserType>, //1-密码 2-指纹 3-MF 卡 4-身份证 7-人脸
    //9- 蓝牙/二维码 90- 蓝牙钥匙扣
    "UserID": <UserID>, //锁用户 ID
    "UserName": "<lock user name>",
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": <time zone>, //设备的时区
    "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.2.15. 清除锁用户 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
  "report": "thing.event.ClearNodeData.post",
  "type": "lockuser",
  "params": {
    "Mode": <Mode> // 0-全部 1-清空密码 2-清空卡片 3-清空指纹
                  9-蓝牙/二维码 90-蓝牙钥匙扣 10-人脸
    "Mode_CN": "<mode name>", //清除模式名称, 与 Accept-Language 相关
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": <time zone>, //设备的时区
    "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.2.16. 设置锁用户名称 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
  "report": "thing.event.SetUserName.post",
  "type": "lockuser",
  "params": {
    "UserID": <UserID>, //UserID
    "UserName": "<UserName>", //(改后)用户名
    "OldUserName": "<old UserName>" //(改前)用户名
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.2.17. 开锁失败 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
  "report": "thing.event.FailToOpenLock.post",
```

```

"type": "notification",
"params": {
  "InvalidType": <type>, //开锁失败类型:
                        0-无效用户 1-过期用户 2-暂停用户 3-开锁密钥失效
  "InvalidType_CN": "<type name>", //本地化开锁失败类型名称, 与 Accept-Language 相关
  "Mode": <mode>, //1-密码 2-指纹 3-卡 4-身份证 5-App 远程 7-人脸 91-二维码
  "Mode_CN": "<mode name>", //开锁模式名称, 与 Accept-Language 相关
  "UserID": <UserID>, //锁用户 id
  "UserName": "<UserName>", //锁用户名
  "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
  "HappenedZone": <time zone>, //设备的时区
  "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
},
"ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.18. NB 锁人工唤醒 report

```

{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": 1, //锁设备在网关设备上的节点 id
  "report": "thing.event.Wakeup.post",
  "type": "notification",
  "params": {
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": <time zone>, //设备的时区
    "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

【注】NB 锁在线状态下人工按*0 唤醒锁后, 锁会上报此事件

3.4.2.19. 获取蓝牙钥匙 report

```

{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
  "report": "thing.event.GetBluetoothKey.post",
  "type": "notification",
  "params": {
    "UserID": <UserID>, //(获取蓝牙钥匙的)用户 id

```

```

        "UserName": "<UserName>"    //(获取蓝牙钥匙的)用户名
    },
    "ts": <timestamp>,                //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>",    //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.20. 更换蓝牙密钥 report

```

{
    "id": <id>,                //report id
    "deviceid": "<deviceid>",   //网关设备 id
    "nodeid": <nodeid>,        //锁设备在网关设备上的节点 id
    "report": "thing.event.UpdateBluetoothSecrets.post",
    "type": "notification",
    "params": {
        "BluetoothSecrets": "<bluetooth secret data>" // (锁生成)蓝牙密钥 TEA 加密数据
        "UserID": <UserID>,                //(更换蓝牙密钥的管理员)用户 id
        "UserName": "<UserName>"           //(更换蓝牙密钥的管理员)用户名
    },
    "ts": <timestamp>,                //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>",    //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.21. 复位蓝牙密钥 report

```

{
    "id": <id>,                //report id
    "deviceid": "<deviceid>",   //网关设备 id
    "nodeid": <nodeid>,        //锁设备在网关设备上的节点 id
    "report": "thing.event.ResetBluetoothSecrets.post",
    "type": "notification",
    "params": {
        "UserID": <UserID>,                //(复位蓝牙密钥的管理员)用户 id
        "UserName": "<UserName>"           //(复位蓝牙密钥的管理员)用户名
    },
    "ts": <timestamp>,                //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>",    //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.22. 异步批量添加/修改锁用户 report

```

{

```

```

    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
    "report": "thing.event.AsyncBatchAddUsers.post",
    "type": "lockuser",
    "params": {
        "UserList": [
            {
                "UserType": <UserType>, //类型: 1-密码 2-指纹 3-MF 卡 4-身份证
                "UserLevel": <UserLevel>, //级别: 1-管理员 2-普通用户 4-临时密码(UserType 必须为 1)
                "UserID": <UserID>, //UserID 规则:
                    管理员: 1~9
                    普通用户: 10~230
                    一次性临时密码: 243~252
                "UserPassword": "<UserPassword>", //密码串或指纹数据串或卡数据
                    卡数据: MF 卡, 4byte 值转换为十六进制串
                    身份证卡, 8byte 值转换为十六进制串
                "ValidWeekDays": <SumWeekdayValue>, //有效星期累加值: 如 5 表示星期天和星期二有效
                    1-星期天 2-星期一 4-星期二 8-星期三
                    16-星期四 32-星期五 64-星期六
                "ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间
                "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
                "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
                "UserName": "<lock user name>"
            },
            ...
        ]
    }
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.23. 异步批量删除锁用户 report

```

{
    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
    "report": "thing.event.AsyncBatchDeleteUsers.post",
    "type": "lockuser",
    "params": {
        "UserIDs": [<userid>, ...] //已删除的锁用户 id 数组
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.24. 锁的附加门磁的最新状态 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "DoorSensorOpened", //属性名称
    "property_cn": "DoorSensorOpened", //本地化属性名称, 与 Accept-Language 相关
    "value": <true/false>, //门磁当前是否是开状态: true-是 false-否
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": <time zone>, //设备的时区
    "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.2.25. 是否允许远程开锁最新状态 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "EnableRemoteOpen", //属性名称
    "property_cn": "EnableRemoteOpen", //本地化属性名称, 与 Accept-Language 相关
    "value": <true/false>, //是否允许远程开锁: true-是 false-否
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": <time zone>, //设备的时区
    "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.2.26. 门铃最新状态 report

```
{
```

```

    "id":<id>, //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>, //设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report":"thing.event.property.post",
    "type":"notification", //notification-通知
    "params":{
        "property": "BellTurnOnStatus", //属性名称
        "property_cn": "BellTurnOnStatus", //本地化属性名称, 与 Accept-Language 相关
        "value": <true/false>, //门铃响铃状态: true-是 false-否
        "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
        "HappenedZone": <time zone>, //设备的时区
        "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.27. (多个)属性状态 report

```

{
    "id":<id>, //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>, //设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report":"thing.event.properties.post",
    "type":"notification", //notification-通知
    "params":{
        "properites":{
            "BluetoothVersion": "<version>",
            "CardNumber": "<card number>",
            "IMEI": "<imei>",
            "LockBoardVersion": "<version>",
            "Network": "<network mode>",
            "Online": true,
            "RSSI": <RSSI percentage>,
            "Version": "<version>"
        }
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>" //本地化事件名称, 与 Accept-Language 相关
}

```

【注】不同类型的锁的 properties 会有所不同, 比如 WIFI 锁会有 SSID 属性、蓝牙锁有 BluetoothVersion 属性。

3.4.2.28. 停用锁用户 report

```

{

```

```

    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
    "report": "thing.event.PauseUser.post",
    "type": "lockuser",
    "params": {
        "UserType": <UserType>, //1-密码 2-指纹 3-MF 卡 4-身份证 7-人脸
        "UserID": <UserID>, //锁用户 ID
        "UserName": "<lock user name>",
        "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
        "HappenedZone": <time zone>, //设备的时区
        "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.29. 恢复锁用户 report

```

{
    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
    "report": "thing.event.RestoreUser.post",
    "type": "lockuser",
    "params": {
        "UserType": <UserType>, //1-密码 2-指纹 3-MF 卡 4-身份证 7-人脸
        "UserID": <UserID>, //锁用户 ID
        "UserName": "<lock user name>",
        "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
        "HappenedZone": <time zone>, //设备的时区
        "Happened_ts": <happened timestamp> //设备事件发生 Unix 时间戳(秒)
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.2.30. 身份证认证结果 report

```

{
    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //锁设备在网关设备上的节点 id
    "report": "thing.event.IDAuthentication.post",
    "params": {
        "UID": "<UID>", //身份卡卡号(HEX)
    }
}

```

```

        "Result": <result>,           //认证结果: 1-成功, 其它值失败
        "Len": <len>,                 //结果信息长度
        "Info": "<json>"              //结果信息
    },
    "ts": <timestamp>,                //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>"     //系统处理事件的日期时间
}

```

【注】

结果信息示例:

```

{"ClientID":"23425550474078","SAMID":"50157046038","State":"IDLE","ErrCode":1000,"Data":{"\xingming\":"张三
\","\xingbie\":"1","\minzu\":"01","\chusheng\":"19000101","\zhuzhi\":"广东省深圳市龙岗区
\","\sfzno\":"410123456789123456","\qfjg\":"深圳市公安局龙华分局
\","\st\":"20100101","\et\":"20990101","\zp\":"\"}}

```

3.4.2.31. 上报密码/卡号 report

```

{
    "id": <id>,                       //report id
    "deviceid": "<deviceid>",          //网关设备 id
    "nodeid": <nodeid>,               //锁设备在网关设备上的节点 id
    "report": "thing.event.Code.post",
    "type": "notification",
    "params": {
        "Type": "<type>",              //类型: password-密码 mfc card-MF 卡 idcard-身份证卡
        "Code": "<HEX>"                //密码或卡号(HEX)
    },
    "ts": <timestamp>,                //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>",     //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

【注】仅支持 z-wave 锁

3.4.2.32. 串口透传 report

```

{
    "id": <id>,                       //report id
    "deviceid": "<deviceid>",          //网关设备 id
    "nodeid": <nodeid>,               //锁设备在网关设备上的节点 id
    "report": "thing.event.Serial.post",
    "type": "notification",
    "params": {
        "Hex": "<HEX>"                //串口透传上报数据(HEX 串)
    },
    "ts": <timestamp>,                //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>",     //系统处理事件的日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

【注】仅部分人脸门禁设备支持

3.4.3. 开关/无源开关的 report

3.4.3.1. 开关最新状态 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //开关在网关设备上的节点 id
  "channelid":<channelid>, //开关的通道 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "Status", //属性名称
    "property_cn": "status", //本地化属性名称, 与 Accept-Language 相关
    "value": "<on/off>" //开关当前状态: on-开状态 off-关状态
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.4. 窗帘的 report

3.4.4.1. 窗帘最新状态 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //开关在网关设备上的节点 id
  "channelid":<channelid>, //开关的通道 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "OpenPercentage", //属性名称
    "property_cn": "open percentage", //本地化属性名称, 与 Accept-Language 相关
    "value": <percent> //窗帘打开的程度(百分比), 0 表示关
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.5. 调光灯的 report

3.4.5.1. 调光灯最新状态 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //开关在网关设备上的节点 id
  "channelid":<channelid>, //开关的通道 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "OpenPercentage", //属性名称
    "property_cn": "open percentage", //本地化属性名称, 与 Accept-Language 相关
    "value": <percent> //调光灯的亮度(百分比), 0 表示关
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.6. 门磁/窗磁的 report

3.4.6.1. 门磁/窗磁最新状态 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //门磁/窗磁设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "Opened", //属性名称
    "property_cn": "opened", //本地化属性名称, 与 Accept-Language 相关
    "value": <true/false> //门磁/窗磁当前是否是开状态: true-是 false-否
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.6.2. 门磁/窗磁电池电量 report

```
{
  "id":<id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid":<nodeid>, //门磁/窗磁设备在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.property.post",
  "type": "notification", //notification-通知
  "params": {
    "property": "BatteryPercentage",
    "property_cn": "battery percentage ", //本地化属性名称, 与 Accept-Language 相关
    "value":<percent> //电池电量百分比: 0~100
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.6.3. 门磁/窗磁电池低电量告警 report

```
{
  "id":<id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid":<nodeid>, //门磁/窗磁设备在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.ElectricityAlarm.post",
  "type": "alarm", //alarm-告警
  "params": {
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.6.4. 门磁/窗磁设备告警 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //门磁/窗磁设备在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.DeviceAlarm.post",
  "type": "alarm", //alarm-告警
  "params": {
    "AlarmType": <AlarmType>, //告警类型: 1-防拆
  }
}
```

```

        "TipInfo": "<TipInfo>" //告警提示信息
    },
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.6.5. (多个)属性状态 report

```

{
    "id":<id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>, //设备在网关设备上的节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.properties.post",
    "type": "notification", //notification-通知
    "params": {
        "properites": {
            "CardNumber": "<card number>",
            "IMEI": "<imei>",
            "Online": <true/false>,
            "RSSI": "<RSSI percentage>",
            "Version": "<version>"
        }
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>" //本地化事件名称, 与 Accept-Language 相关
}

```

【注】新版门磁(V2)没有 Online 属性

3.4.7. 移动感应的 report

3.4.7.1. 进入 report

```

{
    "id":<id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>, //移动感应设备在网关设备上的节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.Movein.post",
    "type": "notification", //notification-通知
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.7.2. 离开 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //移动感应设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.Moveout.post",
  "type":"notification", //notification-通知
  "ts":<timestamp>, //Unix 时间戳(秒)
  "dt":<yyyy-mm-dd HH:MM:SS>, //日期时间
  "report_cn":<report event name>, //本地化事件名称, 与 Accept-Language 相关
  "detail_cn":<report event description> //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.7.3. 移动感应电池电量 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //移动感应设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "BatteryPercentage",
    "property_cn": "battery percentage ", //本地化属性名称, 与 Accept-Language 相关
    "value":<percent> //电池电量百分比: 0~100
  },
  "ts":<timestamp>, //Unix 时间戳(秒)
  "dt":<yyyy-mm-dd HH:MM:SS>, //日期时间
  "report_cn":<report event name>, //本地化事件名称, 与 Accept-Language 相关
  "detail_cn":<report event description> //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.7.4. 移动感应电池低电量告警 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //移动感应设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.ElectricityAlarm.post",
  "type":"alarm", //alarm-告警
  "ts":<timestamp>, //Unix 时间戳(秒)
  "dt":<yyyy-mm-dd HH:MM:SS>, //日期时间
  "report_cn":<report event name>, //本地化事件名称, 与 Accept-Language 相关
}
```

```

    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.7.5. 移动感应设备告警 report

```

{
    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //移动感应设备在网关设备上的节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.DeviceAlarm.post",
    "type": "alarm", //alarm-告警
    "params": {
        "AlarmType": <AlarmType>, //告警类型: 1-防拆
        "TipInfo": "<TipInfo>" //告警提示信息
    },
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.8. 空调的 report

3.4.8.1. 开/关状态 report

```

{
    "id": <id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid": <nodeid>, //空调设备在网关设备上的节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.property.post",
    "type": "notification", //notification-通知
    "params": {
        "property": "Status", //状态
        "property_cn": "status", //本地化属性名称, 与 Accept-Language 相关
        "value": "<on/off>" //空调当前开/关状态: on-开 off-关
    },
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.8.2. 当前室温 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,    //空调设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property":"RoomTemperature", //室温
    "property_cn": "room temperature", //本地化属性名称, 与 Accept-Language 相关
    "value":{
      "Scale":<C/F>, //温度单位: C-摄氏度 F-华氏度
      "Value":<value> //温度值
    }
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.8.3. 当前运行模式 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,    //空调设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property":"Mode", //模式
    "property_cn": "mode", //本地化属性名称, 与 Accept-Language 相关
    "value":<mode> //空调当前模式: auto-自动 hot-制热 cold-制冷 dry-抽湿 fan-送风
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.8.4. 设定温度 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
```

```

"nodeid":<nodeid>,          //空调设备在网关设备上的节点 id
"logicnodeid":<logicnodeid>, //逻辑节点 id
"report":"thing.event.property.post",
"type":"notification",      //notification-通知
"params":{
    "property":"PointTemperature-<mode>", //设定温度, for: hot-制热 cold-制冷 dry-抽湿
    "property_cn": "target temperature",   //本地化属性名称, 与 Accept-Language 相关
    "value":{
        "Scale":<C/F>,    //温度单位: C-摄氏度 F-华氏度
        "Value":<value>   //温度值
    }
},
"ts": <timestamp>,          //Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.8.5. 设定风速模式 report

```

{
    "id":<id>,          //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>,    //空调设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report":"thing.event.property.post",
    "type":"notification",    //notification-通知
    "params":{
        "property":"FanMode", //风速
        "property_cn": "fan mode", //本地化属性名称, 与 Accept-Language 相关
        "value":<fanmode> //当前风速模式: auto-自动 low-低档 medium-中档 high-高档
    },
    "ts": <timestamp>,          //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.8.6. 空调电池电量 report

```

{
    "id":<id>,          //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>,    //空调设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report":"thing.event.property.post",
    "type":"notification",    //notification-通知
    "params":{
        "property": "BatteryPercentage",

```

```

    "property_cn": "battery percentage ", //本地化属性名称, 与 Accept-Language 相关
    "value": "<percent> " //电池电量百分比: 0~100
  },
  "ts": "<timestamp>", //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.8.7. 空调电池低电量告警 report

```

{
  "id": "<id>", //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": "<nodeid>", //空调设备在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.ElectricityAlarm.post",
  "type": "alarm", //alarm-告警
  "ts": "<timestamp>", //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.9. 温度计的 report

3.4.9.1. 温度 report

```

{
  "id": "<id>", //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": "<nodeid>", //温度计设备在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.property.post",
  "type": "notification", //notification-通知
  "sensortype": "temperaturesensor",
  "params": {
    "property": "Temperature",
    "property_cn": "temperature", //本地化属性名称, 与 Accept-Language 相关
    "value": {
      "Scale": "<C/F>", //温度单位: C-摄氏度 F-华氏度
      "Value": "<value>" //温度值
    }
  },
  "ts": "<timestamp>", //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

```
}
```

3.4.9.2. 温度计电池电量 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //温度计设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "BatteryPercentage",
    "property_cn": " battery percentage ", //本地化属性名称, 与 Accept-Language 相关
    "value":<percent> //电池电量百分比: 0~100
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.9.3. 温度计电池低电量告警 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //温度计设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.ElectricityAlarm.post",
  "type":"alarm", //alarm-告警
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.10. 亮度计的 report

3.4.10.1. 亮度 report

```
{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
```

```

"nodeid":<nodeid>,          //亮度计设备在网关设备上的节点 id
"logicnodeid":<logicnodeid>, //逻辑节点 id
"report":"thing.event.property.post",
"type":"notification",      //notification-通知
"sensortype":"luminancesensor",
"params":{
    "property":"Luminance",
    "property_cn": "luminance",          //本地化属性名称, 与 Accept-Language 相关
    "value":{
        "Scale": "<%/Lux>",          //亮度单位: %-百分比 Lux-勒克斯(照明单位)
        "Value": <value>            //亮度值
    }
},
"ts": <timestamp>,          //Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>",    //日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.10.2. 亮度计电池电量 report

```

{
    "id":<id>,                //report id
    "deviceid":<deviceid>,    //网关设备 id
    "nodeid":<nodeid>,        //亮度计设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report":"thing.event.property.post",
    "type":"notification",      //notification-通知
    "params":{
        "property": "BatteryPercentage",
        "property_cn": "battery percentage ", //本地化属性名称, 与 Accept-Language 相关
        "value":<percent>          //电池电量百分比: 0~100
    },
    "ts": <timestamp>,          //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.10.3. 亮度计电池低电量告警 report

```

{
    "id":<id>,                //report id
    "deviceid":<deviceid>,    //网关设备 id
    "nodeid":<nodeid>,        //亮度计设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report":"thing.event.ElectricityAlarm.post",
    "type":"alarm",           //alarm-告警
    "ts": <timestamp>,          //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间

```

```

    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.11. 湿度计的 report

3.4.11.1. 湿度 report

```

{
    "id":<id>, //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>, //湿度计设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report":"thing.event.property.post",
    "type":"notification", //notification-通知
    "sensortype":"humiditysensor",
    "params":{
        "property":"Humidity",
        "property_cn": "humidity", //本地化属性名称, 与 Accept-Language 相关
        "value":{
            "Scale": "<%/g/m3>", //湿度单位: %-百分比 g/m3-绝对湿度
            "Value": <value> //湿度值
        }
    },
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.11.2. 湿度计电池电量 report

```

{
    "id":<id>, //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>, //湿度计设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report":"thing.event.property.post",
    "type":"notification", //notification-通知
    "params":{
        "property": "BatteryPercentage",
        "property_cn": "battery percentage", //本地化属性名称, 与 Accept-Language 相关
        "value":<percent> //电池电量百分比: 0~100
    },
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
}

```

```

    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.11.3. 湿度计电池低电量告警 report

```

{
    "id":<id>, //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>, //湿度计设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report": "thing.event.ElectricityAlarm.post",
    "type": "alarm", //alarm-告警
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.12. 烟感的 report

3.4.12.1. 烟雾报警 report

```

{
    "id":<id>, //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>, //烟感设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "params": {
        "AlarmType": 1,
        "TipInfo": "smoke alarm"
    },
    "report": "thing.event.SensorAlarm.post",
    "type": "alarm",
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.12.2. 撤销烟雾报警 report

```

{
    "id":<id>, //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>, //烟感设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id

```

```

"params": {
    "AlarmType": 1,
    "TipInfo": "smoke alarm"
},
"report": "thing.event.SensorCancelAlarm.post",
"type": "alarm",
"ts": <timestamp>, //Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.12.3. 烟感电池电量 report

```

{
    "id":<id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>, //烟感设备在网关设备上的节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.property.post",
    "type": "notification", //notification-通知
    "params": {
        "property": "BatteryPercentage",
        "property_cn": "battery percentage ", //本地化属性名称, 与 Accept-Language 相关
        "value":<percent> //电池电量百分比: 0~100
    },
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.12.4. 烟感电池低电量告警 report

```

{
    "id":<id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>, //烟感设备在网关设备上的节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.ElectricityAlarm.post",
    "type": "alarm", //alarm-告警,
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.12.5. 设备告警 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //烟感设备在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.DeviceAlarm.post",
  "type": "alarm", //alarm-告警
  "params": {
    "AlarmType": <AlarmType>, //告警类型: 1-防拆
    "TipInfo": "<TipInfo>" //告警提示信息
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.12.6. 撤销设备告警 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //烟感设备在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.DeviceCancelAlarm.post",
  "type": "alarm", //alarm-告警
  "params": {
    "AlarmType": <AlarmType>, //告警类型: 1-防拆
    "TipInfo": "<TipInfo>" //告警提示信息
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.13. 漏水感应的 report

3.4.13.1. 漏水报警 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //漏水感应设备在网关设备上的节点 id
```

```

"logicnodeid": "<logicnodeid>", //逻辑节点 id
"params": {
    "AlarmType": 5,
    "TipInfo": "water alarm"
},
"report": "thing.event.SensorAlarm.post",
"type": "alarm",
"ts": <timestamp>, //Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.13.2. 撤销漏水报警 report

```

{
    "id":<id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>, //漏水感应设备在网关设备上的节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "params": {
        "AlarmType": 5,
        "TipInfo": "water alarm"
    },
    "report": "thing.event.SensorCancelAlarm.post",
    "type": "alarm",
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.13.3. 漏水感应电池电量 report

```

{
    "id":<id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>, //漏水感应设备在网关设备上的节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.property.post",
    "type": "notification", //notification-通知
    "params": {
        "property": "BatteryPercentage",
        "property_cn": "battery percentage ", //本地化属性名称, 与 Accept-Language 相关
        "value":<percent> //电池电量百分比: 0~100
    },
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.13.4. 漏水感应电池低电量告警 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //漏水感应设备在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.ElectricityAlarm.post",
  "type": "alarm", //alarm-告警,
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.13.5. 设备告警 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //漏水感应设备在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.DeviceAlarm.post",
  "type": "alarm", //alarm-告警
  "params": {
    "AlarmType": <AlarmType>, //告警类型: 1-防拆
    "TipInfo": "<TipInfo>" //告警提示信息
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.13.6. 撤销设备告警 report

```
{
  "id": <id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": <nodeid>, //漏水感应设备在网关设备上的节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.DeviceCancelAlarm.post",
  "type": "alarm", //alarm-告警
  "params": {
    "AlarmType": <AlarmType>, //告警类型: 1-防拆
    "TipInfo": "<TipInfo>" //告警提示信息
  }
}
```

```

    },
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.14. 电表的 report

3.4.14.1. 电表读数 report

```

{
    "id":<id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>, //电表在网关设备上的节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.property.post",
    "type": "notification", //notification-通知
    "params": {
        "property": "KWh",
        "property_cn": "kwh", //本地化属性名称, 与 Accept-Language 相关
        "value": <value> //电表当前读数
    },
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.15. 水表的 report

3.4.15.1. 水表读数 report

```

{
    "id":<id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>, //水表在网关设备上的节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.property.post",
    "type": "notification", //notification-通知
    "params": {
        "property": "CubicMeters",
        "property_cn": "cubic meters", //本地化属性名称, 与 Accept-Language 相关
        "value": <value> //水表当前读数
    },
}

```

```

"ts": <timestamp>, //Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.16. (中科奥森)人脸识别设备的 report

3.4.16.1. 通行 report

```

{
  "id":<id>, //report id
  "deviceid":"<deviceid>", //网关设备 id
  "nodeid":<nodeid>, //水表在网关设备上的节点 id
  "logicnodeid":"<logicnodeid>", //逻辑节点 id
  "report":"thing.event.FaceRecognition.post",
  "type":"notification", //notification-通知
  "params":{
    "ip": "<device ip>", //设备 ip
    "licenseCode": "<device license>", //设备 license code
    "id": <user id>, //人员的 user id
    "name": "<name>", //人员的姓名
    "number": "<number>", //人员的证件号码
    "compareThreshold": <compare threshold>, //比对阈值
    "similarity": <similarity>, //比对相似度
    "veiryTime": "<yyyy-mm-dd HH:MM:SS>" //通行日期时间
  },
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.17. 情景面板的 report

3.4.17.1. 情景面板触控 report

```

{
  "id":<id>, //report id
  "deviceid":"<deviceid>", //网关设备 id
  "nodeid":<nodeid>, //情景面板在网关设备上的节点 id
  "logicnodeid":"<logicnodeid>", //逻辑节点 id
  "report":"thing.event.ScenepanelTap.post",
  "type":"notification", //notification-通知
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
}

```

```

    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.18. 净水器的 report

3.4.18.1. 滤芯剩余时间 report

```

{
    "id":<id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>, //节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.property.post",
    "params": {
        "property": "RemainingTimes",
        "property_cn": "滤芯剩余时间", //本地化属性名称, 与 Accept-Language 相关
        "value": [<value1>, <value2>, <value3>, <value4>, <value5>] //值
    },
    "type": "notification", //notification-通知
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.18.2. 制水状态 report

```

{
    "id":<id>, //report id
    "deviceid": "<deviceid>", //网关设备 id
    "nodeid":<nodeid>, //节点 id
    "logicnodeid": "<logicnodeid>", //逻辑节点 id
    "report": "thing.event.MakeWaterStatus.post",
    "params": {
        "Sid": <sid>, //制水业务 id
        "Status": <status> //制水状态
    },
    "type": "notification", //notification-通知
    "ts": <timestamp>, //Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
    "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
    "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.18.3. 制水记录 report

```
{
  "id":<id>,           //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,    //节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.MakeWaterRecord.post",
  "params":{
    "Sid": <sid>,           //制水业务 id
    "InWaterQuality": <value>, //入水水质
    "OutWaterQuality": <value>, //出水水质
    "MakeWaterTime": <value>, //制水时长
    "MakeWaterTotal": <value> //制水量
  },
  "type":"notification", //notification-通知
  "ts": <timestamp>,      //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.19. SOS 紧急求救设备的 report

3.4.19.1. 电池电量 report

```
{
  "id":<id>,           //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,    //SOS 设备在网关设备上的节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "BatteryPercentage",
    "property_cn": "battery percentage ", //本地化属性名称, 与 Accept-Language 相关
    "value":<percent> //电池电量百分比: 0~100
  },
  "ts": <timestamp>,      //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.19.2. 电池低电量告警 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,   //SOS 设备在网关设备上的节点 id
  "report":"thing.event.ElectricityAlarm.post",
  "type":"alarm",      //alarm-告警
  "ts":<timestamp>,    //Unix 时间戳(秒)
  "dt":<yyyymmddHHMMSS>, //日期时间
  "report_cn":<report event name>, //本地化事件名称, 与 Accept-Language 相关
  "detail_cn":<report event description> //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.19.3. 紧急求救告警 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,   //SOS 设备在网关设备上的节点 id
  "report":"thing.event.SOS.post",
  "type":"alarm",      //alarm-告警
  "ts":<timestamp>,    //Unix 时间戳(秒)
  "dt":<yyyymmddHHMMSS>, //日期时间
  "report_cn":<report event name>, //本地化事件名称, 与 Accept-Language 相关
  "detail_cn":<report event description> //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.20. 洗衣机/洗鞋机/干衣机的 report

3.4.20.1. 刷卡记录 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,   //节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.ScanCardRecord.post",
  "params":{
    "CardNumber":<cardnumber>, //卡号
    "WorkMode":<mode> //工作模式: 1,2,...
  },
  "type":"notification", //notification-通知
  "ts":<timestamp>,    //Unix 时间戳(秒)
  "dt":<yyyymmddHHMMSS> //日期时间
}
```

3.4.21. 串口设备的 report

3.4.21.1. 串口 report

```
{
  "id":<id>, //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid":<nodeid>, //节点 id
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.Serial.post",
  "params": {
    "Index": <index>, //流水号
    "IRData": "<hex>" //串口设备的上报数据, HEX 格式
  },
  "type": "notification", //notification-通知
  "ts": <timestamp>, //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>" //日期时间
}
```

3.4.22. 蓝牙耳机标的 report

3.4.22.1. (多个)属性状态 report

```
{
  "deviceid": "<mac>", //设备 id(耳标设备的 MAC 地址)
  "nodeid":<nodeid>, //节点 id(固定值 1)
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "report": "thing.event.properties.post",
  "type": "notification", //notification-通知
  "params": {
    "properites": {
      "BatteryPercentage": <battery percentage>, //电量(%)
      "RSSI": <RSSI>, //蓝牙信号强度(单位 dBm)
      "Steps": <step count>, //步数
      "Temperature": <temperature> //温度(摄氏度)
    }
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>" //系统处理事件的日期时间
}
```

3.4.23. 燃气感应的 report

3.4.23.1. 燃气报警 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,    //燃气设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "params":{
    "AlarmType": 2,
    "TipInfo": "co alarm"
  },
  "report": "thing.event.SensorAlarm.post",
  "type": "alarm",
  "ts":<timestamp>,          //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.23.2. 撤销燃气报警 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,    //燃气设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "params":{
    "AlarmType": 2,
    "TipInfo": "co alarm"
  },
  "report": "thing.event.SensorCancelAlarm.post",
  "type": "alarm",
  "ts":<timestamp>,          //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.24. 路灯的 report

3.4.24.1. (设备网络状态) 多属性状态 report

```
{
```

```

    "id":<id>, //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>, //设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report": "thing.event.properties.post",
    "type": "notification", //notification-通知
    "params": {
        "properites": {
            "IMEI": "<imei>", //IMEI
            "CardNumber": "<card number>", //IMSI
            "RSSI": <RSSI>, //RSSI
            "SNR": <SNR>, //SNR
            "Version": "<version>" //Version
        }
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>" //本地化事件名称，与 Accept-Language 相关
}

```

3.4.24.2. （设备工作状态）多属性状态 report

```

{
    "id":<id>, //report id
    "deviceid":<deviceid>, //网关设备 id
    "nodeid":<nodeid>, //设备在网关设备上的节点 id
    "logicnodeid":<logicnodeid>, //逻辑节点 id
    "report": "thing.event.properties.post",
    "type": "notification", //notification-通知
    "params": {
        "properites": {
            "Status": "<status>", //当前开关状态: on-开状态 off-关状态
            "Lightness": <lightness>, //路灯的亮度程度(1~100%)
            "Voltage": <value>, //路灯的电压
            "Current": <value>, //路灯的电流
            "Power":<value> //路灯的功率
        }
    },
    "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
    "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
    "report_cn": "<report event name>" //本地化事件名称，与 Accept-Language 相关
}

```

3.4.25. 医药箱的 report

3.4.25.1. （设备版本网络）多属性状态 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,   //设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.properties.post",
  "type":"notification", //notification-通知
  "params":{
    "properites":{
      "IMEI": "<imei>",          //IMEI
      "CardNumber": "<card number>", //IMSI
      "RSSI": <RSSI>,          //RSSI
      "SNR": <SNR>,            //SNR
      "Version": "<version>"    //Version
    }
  },
  "ts": <timestamp>,          //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>" //本地化事件名称，与 Accept-Language 相关
}
```

3.4.25.2. （设备工作状态）多属性状态 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,   //设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.properties.post",
  "type":"notification", //notification-通知
  "params":{
    "properites":{
      "Status": <value>,          //当前开关状态: 1-开状态 0-关状态
      "Temperature": <value>,    //当前温度
      "PointTemperature": <value> //设定温度
    }
  },
  "ts": <timestamp>,          //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>" //本地化事件名称，与 Accept-Language 相关
}
```

3.4.25.3. (设备地理位置) 多属性状态 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,   //设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.properties.post",
  "type":"notification", //notification-通知
  "params":{
    "properites":{
      "Longitude":<value>, //经度
      "Latitude":<value>, //纬度
      "Altitude":<value> //海拔
    }
  },
  "ts":<timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt":<yyyy-mm-dd HH:MM:SS>, //系统处理事件的日期时间
  "report_cn":<report event name> //本地化事件名称, 与 Accept-Language 相关
}
```

3.4.25.4. 电池电量 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>,   //节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "BatteryPercentage",
    "property_cn": "battery percentage ", //本地化属性名称, 与 Accept-Language 相关
    "value":<percent>, //电池电量百分比: 0~100
    "HappenedTime":<YYYY-MM-DD HH:MM>, //设备事件发生时间(设备的本地时间)
    "HappenedZone":<time zone>, //设备的时区
    "Happened_ts":<happened timestamp> //设备事件发生 Unix 时间戳(秒)
  },
  "ts":<timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt":<yyyy-mm-dd HH:MM:SS>, //系统处理事件的日期时间
  "report_cn":<report event name>, //本地化事件名称, 与 Accept-Language 相关
  "detail_cn":<report event description> //本地化事件描述, 与 Accept-Language 相关
}
```

3.4.25.5. 开箱 report

```
{
  "id":<id>,          //report id
  "deviceid":<deviceid>, //网关设备 id
```

```

"nodeid":<nodeid>,          //节点 id
"report":"thing.event.DoorOpenNotification.post",
"type":"notification",      //notification-通知
"params":{
  "Mode": 9,                //9- 蓝牙
  "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
  "HappenedZone": <time zone>,          //设备的时区
  "Happened_ts": <happened timestamp>    //设备事件发生 Unix 时间戳(秒)
},
"ts": <timestamp>,          //系统处理事件的 Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>",      //系统处理事件的日期时间
"report_cn": "<report event name>",  //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.25.6. 附加门磁的最新状态 report

```

{
  "id":<id>,                //report id
  "deviceid":<deviceid>,    //网关设备 id
  "nodeid":<nodeid>,        //节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report":"thing.event.property.post",
  "type":"notification",     //notification-通知
  "params":{
    "property": "DoorSensorOpened", //属性名称
    "property_cn": "DoorSensorOpened", //本地化属性名称, 与 Accept-Language 相关
    "value": <true/false>,          //门磁当前是否是开状态: true-是 false-否
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": <time zone>,          //设备的时区
    "Happened_ts": <happened timestamp>    //设备事件发生 Unix 时间戳(秒)
  },
  "ts": <timestamp>,          //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>",      //系统处理事件的日期时间
  "report_cn": "<report event name>",  //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.26. 电梯的 report

3.4.26.1. 乘梯 report

```

{
  "id":<id>,                //report id
  "deviceid":<deviceid>,    //网关设备 id
  "nodeid":<nodeid>,        //设备在 DTU 设备上的节点 id
  "report":"thing.event.Lift.post",
  "type":"notification",     //notification-通知

```

```

"params":{
  "Mode": "<mode>", //梯控方式: 2-指纹 3-卡联网 5-App 远程 7-人脸 8-卡脱网 91-二维码
  "UserID": "<userid>", //用户 id
  "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
  "HappenedZone": "<time zone>", //设备的时区
  "Happened_ts": "<happened timestamp>" //设备事件发生 Unix 时间戳(秒)
},
"ts": "<timestamp>", //系统处理事件的 Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.26.2. 无效乘梯 report

```

{
  "id": "<id>", //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": "<nodeid>", //设备在 DTU 设备上的节点 id
  "report": "thing.event.FailToLift.post",
  "type": "notification",
  "params": {
    "InvalidType": "<type>", //失败类型:
                                0-无效用户 1-过期用户 2-暂停用户 3-重复卡 4-次数限制
    "InvalidType_CN": "<type name>", //失败类型名称, 与 Accept-Language 相关
    "Mode": "<mode>", //梯控方式: 2-指纹 3-卡联网 7-人脸 8-卡脱网 91-二维码
    "Mode_CN": "<mode name>", //梯控方式名称, 与 Accept-Language 相关
    "UserID": "<UserID>", //电梯用户 id
    "Code": "<HEX>", //卡号(HEX)
    "HappenedTime": "<YYYY-MM-DD HH:MM>", //设备事件发生时间(设备的本地时间)
    "HappenedZone": "<time zone>", //设备的时区
    "Happened_ts": "<happened timestamp>" //设备事件发生 Unix 时间戳(秒)
  },
  "ts": "<timestamp>", //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.26.3. 上报卡号 report

```

{
  "id": "<id>", //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": "<nodeid>", //设备在 DTU 设备上的节点 id
  "report": "thing.event.Code.post",
  "type": "notification",
  "params": {
    "Type": "<type>", //类型: mfcard-MF 卡 idcard-身份证卡
    "Code": "<HEX>" //卡号(HEX)
  },
}

```

```

"ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

3.4.27. 电梯对讲设备的 report

3.4.27.1. (设备上电)多属性状态 report

```

{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report": "thing.event.properties.post",
  "type": "notification", //notification-通知
  "params": {
    "properites": {
      "IMEI": "869728062899436",
      "Model": "S1000",
      "RSSI": 28,
      "Version": "v1.0"
    }
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>" //本地化事件名称, 与 Accept-Language 相关
}

```

3.4.27.2. (设备 IP 定位)多属性状态 report

```

{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //设备在网关设备上的节点 id
  "logicnodeid":<logicnodeid>, //逻辑节点 id
  "report": "thing.event.properties.post",
  "type": "notification", //notification-通知
  "params": {
    "properties": {
      "CityCode": "440300", //市编码
      "CityName": "深圳市", //市名称
      "IP": "113.110.172.71", //公网 IP
      "Latitude": 22.547, //纬度
      "Longitude": 114.085947, //经度
      "ProvinceCode": "440000", //省编码
      "ProvinceName": "广东省" //省名称
    }
  }
}

```

```

    }
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //系统处理事件的日期时间
  "report_cn": "<report event name>" //本地化事件名称, 与 Accept-Language 相关
}

```

【注】以下情况下会产生该 report:

Case A: 首次 online, 或者 offline 后再 online;

Case B: 设备的公网 IP 发生变化

3.4.28. 断路器的 report

3.4.28.1. 最新状态 report

```

{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //锁设备在网关设备上的节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "Status",
    "value": "<on/off>" //on-合闸状态 off-分闸状态
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>" //系统处理事件的日期时间
}

```

3.4.28.2. 功耗读数 report

```

{
  "id":<id>, //report id
  "deviceid":<deviceid>, //网关设备 id
  "nodeid":<nodeid>, //锁设备在网关设备上的节点 id
  "report":"thing.event.property.post",
  "type":"notification", //notification-通知
  "params":{
    "property": "Kwh",
    "value": "<value>"
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>" //系统处理事件的日期时间
}

```

3.4.28.3. 电压电流功率 report

```
{
  "id":<id>,          //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid":<nodeid>,   //锁设备在网关设备上的节点 id
  "report": "thing.event.property.post",
  "type": "notification", //notification-通知
  "params": {
    "property": "<AVCP/BVCP/CVCP>", //AVCP-A 厢 BVCP-B 厢 CVCP-C 厢
    "value": {
      "C": {
        "Scale": "A",          //电流
        "Value": "<value>"
      },
      "P": {
        "Scale": "W",          //功率
        "Value": "<value>"
      },
      "V": {
        "Scale": "V",          //电压
        "Value": "<value>"
      }
    }
  },
  "ts": <timestamp>,      //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>" //系统处理事件的日期时间
}
```

3.4.28.4. 温度 report

```
{
  "id":<id>,          //report id
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid":<nodeid>,   //锁设备在网关设备上的节点 id
  "report": "thing.event.property.post",
  "type": "notification", //notification-通知
  "params": {
    "property": "Temperature",
    "value": {
      "Scale": "C",
      "Value": "<value>"
    }
  },
  "ts": <timestamp>,      //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>" //系统处理事件的日期时间
}
```

3.4.28.5. 用电安全告警 report

```
{
  "deviceid": "<deviceid>", //网关设备 id
```

```

"nodeid": "<nodeid>", //断路器设备在网关设备上的节点 id
"report": "thing.event.DeviceAlarm.post",
"type": "alarm", //alarm-告警
"params": {
  "AlarmType": [<AlarmType>,<AlarmType>,...] //告警类型 ID(1 或 N 项)
  "TipInfo": "<TipInfo>" //告警信息(多项,分隔)
},
"ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>" //系统处理事件的日期时间
}

```

3.4.28.6. 用电安全撤销告警 report

```

{
  "deviceid": "<deviceid>", //网关设备 id
  "nodeid": "<nodeid>", //断路器设备在网关设备上的节点 id
  "report": "thing.event.DeviceCancelAlarm.post",
  "type": "alarm", //alarm-告警
  "params": {
    "AlarmType": 0,
    "TipInfo": "全部"
  },
  "ts": <timestamp>, //系统处理事件的 Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>" //系统处理事件的日期时间
}

```

3.5. 接收异步请求的操作结果

url: ws://<ip>:<port>/ws/receiveAsyncReply

encoding: UTF-8

*生产环境使用 wss

【注】异步请求参见：

[添加/修改锁用户](#)

[删除锁用户](#)

[修改锁用户的有效日期时间](#)

[修改锁用户的有效星期](#)

[远程开锁](#)

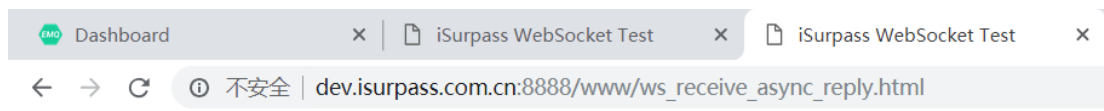
[远程落锁](#)

[添加卡](#)

[添加指纹](#)

javascript client 示例：

[http\(s\)://112.74.20.113:8888/www/ws_receive_async_reply.html](http(s)://112.74.20.113:8888/www/ws_receive_async_reply.html)



receiveAsyncReply Socket is open

async invoke reply: {"resultCode":1,"receipt":"wifilock-1559115891459"}

3.6. 请求指纹采集器采集指纹

url: ws://<ip>:<port>/ws/fingerprint

encoding: UTF-8

*生产环境使用 wss

1) 请求指纹采集器进入采集指纹状态:

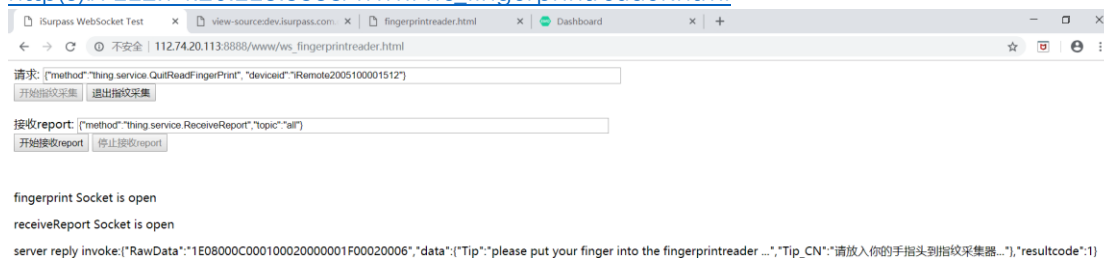
{"method":"thing.service.EnterReadFingerPrint", "deviceid":"<deviceid>"}

2) 请求指纹采集器退出采集指纹状态:

{"method":"thing.service.QuitReadFingerPrint", "deviceid":"<deviceid>"}

示例:

[http\(s\)://112.74.20.113:8888/www/ws_fingerprintreader.html](http(s)://112.74.20.113:8888/www/ws_fingerprintreader.html)



请求

DeviceID: jSurpass2019100040

NodeID: 2

FireWare FileName: DoorLock15_bee_ota_305.bin

开始升级

```
OTA Socket is open
receiveReport Socket is open

server reply invoke:{"RawData":"1E08000E00470002000200460004800A0000","params":{"TiplInfo":"the device has received data package 1"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A0001","params":{"TiplInfo":"the device has received data package 2"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A0002","params":{"TiplInfo":"the device has received data package 3"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A0003","params":{"TiplInfo":"the device has received data package 4"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A0004","params":{"TiplInfo":"the device has received data package 5"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A0005","params":{"TiplInfo":"the device has received data package 6"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A0006","params":{"TiplInfo":"the device has received data package 7"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A0007","params":{"TiplInfo":"the device has received data package 8"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A0008","params":{"TiplInfo":"the device has received data package 9"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A0009","params":{"TiplInfo":"the device has received data package 10"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A000A","params":{"TiplInfo":"the device has received data package 11"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A000B","params":{"TiplInfo":"the device has received data package 12"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A000C","params":{"TiplInfo":"the device has received data package 13"},"resultcode":1}

server reply invoke:{"RawData":"1E08000E00470002000200460004800A02A6","params":{"TiplInfo":"the device has received data package 679"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02A7","params":{"TiplInfo":"the device has received data package 680"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02A8","params":{"TiplInfo":"the device has received data package 681"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02A9","params":{"TiplInfo":"the device has received data package 682"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02AA","params":{"TiplInfo":"the device has received data package 683"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02AB","params":{"TiplInfo":"the device has received data package 684"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02AC","params":{"TiplInfo":"the device has received data package 685"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02AD","params":{"TiplInfo":"the device has received data package 686"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02AE","params":{"TiplInfo":"the device has received data package 687"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02AF","params":{"TiplInfo":"the device has received data package 688"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02B0","params":{"TiplInfo":"the device has received data package 689"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02B1","params":{"TiplInfo":"the device has received data package 690"},"resultcode":1}
server reply invoke:{"RawData":"1E08000E00470002000200460004800A02B2","params":{"TiplInfo":"the device has received data package 691"},"resultcode":1}
server reply invoke:{"RawData":"1E08000D00470002000200460003800B00","params":{"TiplInfo":"device checksum data success"},"resultcode":1}
server push report:{"RawData":"1E09000D00470002000200460003800C00","deviceid":"jSurpass2019100040","id":0,"loginodeid":"jSurpass2019100040-2-0-","nodeid":2,"params":{"Result":1,"TiplInfo":"device OTA upgrade success"},"report":{"thing.event.OTA.post","type":"notification"}}
```

3.8. 远程控制锁

url: ws://<ip>:<port>/ws/lock

encoding: UTF-8

*生产环境使用 wss

1) 远程控制锁进入增加卡用户:

```
{"method":"thing.service.EnterAddCard", "async": false, "deviceid":"<deviceid>","nodeid":"<nodeid>"}
```

【注】

NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1

2) 远程控制锁进入增加指纹用户:

```
{"method":"thing.service.EnterAddFingerPrint", "async": false, "deviceid":"<deviceid>","nodeid":"<nodeid>"}
```

【注】

*开发环境可以用 http, 生产环境建议使用 https

4.1.1. 设置网关扩展属性

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetDeviceExtendedAttribute",
  "deviceid": "<deviceid>", //网关设备 ID
  "params": {
    "<Name>": <value> //扩展属性名: 属性值
  }
}
```

【注】属性名可以是任何字母打头的字符串, 属性值可以是任何 json 值(字符串, 整型, 浮点型, 布尔, 对象, 数组)

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://localhost/app \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'Content-Type: application/json' \
  -H 'Postman-Token: 406f8f51-5bed-476e-ac82-dc0688df3b98' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.SetDeviceExtendedAttribute",
    "deviceid": "iRemote7005000000020",
    "params": {
      "Name": "MQTT 网关"
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

4.1.2. 获取网关属性

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetDeviceAttribute",
  "deviceid": "<deviceid>",          //网关设备 ID
  "params": {
    "WithoutProperties": <true/false>  //不获取设备状态属性, 缺省 false
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
  "data": {
    "Name": "<value>",        //扩展属性名: 属性值
    ...
    "_deviceid": "<deviceid>", //内置的设备属性
    "_isvirtual": true,        //是否是虚拟设备
    "_supportdsk": <true/false>, //是否支持 DSK 设备注册
    "_properties": {
      "Online": <true/false>,   //最新状态: true-在线 false-离线
      "LastestDateTime": "<yyyy-mm-dd HH:MM:SS>", //最新状态日期时间
      "Lastest_ts": <timestamp>, //最新状态 Unix 时间戳(毫秒)
      "IP": "<ip>",             //设备的互联网 IP
      "Network": "<network type>", //网络类型: LAN、WIFI、2G
      "Version": "<software version>", //设备的软件版本号
      "SSID": "<wifi ssid>",      //WIFI SSID
      "WIFI_IP": "<wifi ip>",     //WIFI IP
      "WIFI_MAC": "<wifi mac>",   //WIFI MAC
      "RSSI": <value>           //wifi 或蜂窝移动信号强度值
    }
  }
}
```

【注】

1) _打头的属性是系统内置的设备属性

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'Content-Type: application/json' \
  -d '{
    "method": "thing.service.GetDeviceAttribute",
    "deviceid": "iRemote7005000000020"
  }'
```

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "_deviceid": "iRemote7005000000020",
    "_properties": {
      "IP": "119.123.42.149",
    }
  }
}
```

```

        "LastestDateTime": "2019-07-09 11:52:53",
        "Lastest_ts": 1562644373,
        "Network": "WIFI",
        "Online": true,
        "RSSI": 80,
        "SSID": "Tplink_Office_2.4G",
        "Version": "v1.0.20",
        "WIFI_IP": "192.168.5.6",
        "WIFI_MAC": "F0FE6B50B7E3"
    }
}
}

```

4.1.3. 按扩展属性查询网关

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.GetDeviceList",
    "params": {
        "WithoutProperties": <true/false>, //不获取设备状态属性, 缺省 false
        "Conditions": {"<attr>":"<value>"}, //属性:值, 查询条件可以指定多个(与的关系)
        "SortBy": ["<attr>"], //排序属性, attr 升序, -attr 降序, 排序可以指定多个
        "PageSize": <pagesize>, //每页记录数
        "Page": <page> //第几页
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功, 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
    "data": {
        "TotalCount": <TotalCount> //总记录数
        "PageCount": <PageCount> //总页数
        "PageSize": <pagesize>, //每页记录数
        "Page": <page>, //当前第几页
        "PageList": [ //当前页的设备列表
            {
                "Name": "<value>", //属性名: 属性值
                ...
                "_deviceid": "<deviceid>", //内置的设备属性
                "_isvirtual": true, //是否是虚拟设备
                "_supportdsk": <true/false>, //是否支持 DSK 设备注册
                "_properties": {
                    "Online": <true/false>, //最新状态: true-在线 false-离线
                    "LastestDateTime": "<yyyy-mm-dd HH:MM:SS>", //最新状态日期时间
                    "Lastest_ts": <timestamp>, //最新状态 Unix 时间戳(毫秒)
                }
            }
        ]
    }
}

```

```

        "IP": "<ip>", //设备的互联网 IP
        "Network": "<network type>", //网络类型：LAN、WIFI、2G
        "Version": "<software version>", //设备的软件版本号
        "SSID": "<wifi ssid>", //WIFI SSID
        "WIFI_IP": "<wifi ip>", //WIFI IP
        "WIFI_MAC": "<wifi mac>", //WIFI MAC
        "RSSI": "<value>" //wifi 或蜂窝移动信号强度值
    }
    ...
}
...
}
]
}

```

【注】

1) Z-Wave 网关无 `isvirtual` 属性，WIFI 锁/NB 锁/指纹采集器等设备系统会自动生成一个虚拟网关：
`_isvirtual` 属性值为 true

2) 该接口不适合于以下使用方式：
业务应用系统周期性地全量同步设备及其状态

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/app \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'Content-Type: application/json' \
  -H 'Postman-Token: a4ebcb91-0832-4280-affe-49627626946a' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.GetDeviceList",
    "params": {
      "SortBy": ["Name"]
    }
  }'

```

//返回结果：

```

{
  "resultcode": 1,
  "data": {
    "TotalCount": 1,
    "PageCount": 1,
    "PageSize": 10,
    "Page": 1,
    "PageList": [
      {
        "_deviceid": "iRemote7005000000020",
        "_properties": {
          "IP": "119.123.42.149",
          "LastestDateTime": "2019-07-09 11:52:53",
          "Lastest_ts": 1562644373,
          "Network": "WIFI",
          "Online": true,
          "RSSI": 80,
          "SSID": "Tplink_Office_2.4G",
          "Version": "v1.0.20",
          "WIFI_IP": "192.168.5.6",

```

```

        "WIFI_MAC": "F0FE6B50B7E3"
    }
}
]
}
}

```

4.1.4. 设置节点设备扩展属性

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.SetNodeExtendedAttribute",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": <nodeid>,                 //智能设备在网关设备上的节点 id
  "params": {
    "<Name>": <value>                 //扩展属性名: 属性值
  }
}

```

【注】属性名可以是任何字母打头的字符串，属性值可以是任何 json 值(字符串，整型，浮点型，布尔，对象，数组)

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,           //1-成功 非 1-失败
  "errorinfo": "",           //错误信息
  "errorinfo_cn": ""         //本地化错误信息(Accept-Language 相关)
}

```

[示例]

```

curl -X POST \
  http(s)://localhost/app \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'Content-Type: application/json' \
  -H 'Postman-Token: 406f8f51-5bed-476e-ac82-dc0688df3b98' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.SetNodeExtendedAttribute",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params": {
      "Name": "test lock2",
      "SSID": "test"
    }
  }'

```

//返回结果:

```

{
  "resultcode": 1
}

```

```
}
```

4.1.5. 获取节点设备属性

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.GetNodeAttribute",
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid": "<nodeid>",                  //智能设备在网关设备上的节点 id
  "params": {
    "WithoutProperties": <true/false>    //不获取设备状态属性, 缺省 false
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                      //1-成功 非 1-失败
  "errorinfo": "",                      //错误信息
  "errorinfo_cn": "",                  //本地化错误信息(Accept-Language 相关)
  "data": {
    "Name": "<value>",                  //扩展属性名: 属性值
    ...
    "_deviceid": "<deviceid>",          //内置的设备属性
    "_isentity": <true/false>,         //内置的设备属性
    "_nodeid": "<nodeid>",              //内置的设备属性
    "_nodetype": "<devicetype>",        //内置的设备属性
    "_nodeprotocoltype": "<nodeprotocoltype>", //内置的设备属性
    "_islogic": <true/false>,          //内置的设备属性
    "_logicid": "<logicid>",           //内置的设备属性
    "_logicnodetype": "<logicnodetype>", //内置的设备属性
    "_logicnodeprotocoltype": "<logicnodeprotocoltype>", //内置的设备属性
    "_isbluetooth": true,              //内置的设备属性
    "_bluetooth_mac": "<bluetooth MAC>", //内置的设备属性
    "_bluetooth_name": "<bluetooth name>", //内置的设备属性
    "_dsk": "",                        //内置的设备属性
    "properties": {                    //内置的设备属性
      "Name": "<value>"                //属性名: 属性值
      ...
    }
  }
}
```

【注】

- 1) _打头的属性是系统内置的设备属性
- 2) 仅蓝牙设备有_isbluetooth、_bluetooth_mac、_bluetooth_name 这三个属性

[示例]

```
curl -X POST \
```

```

http(s)://112.74.20.113:8888/app \
-H 'Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
-H 'Content-Type: application/json' \
-H 'Postman-Token: 043d0802-d7a1-4ff9-b7e4-eb005dcca176' \
-H 'cache-control: no-cache' \
-d '{
"method": "thing.service.GetNodeAttribute",
"deviceid": "iSurpass2018100000",
"nodeid": 2
}'
//返回结果:
{
  "resultcode": 1,
  "data": {
    "Name": "test lock2",
    "SSID": "test",
    "_deviceid": "iSurpass2018100000",
    "_isentity": true,
    "_islogic": true,
    "_logicid": "iSurpass2018100000-2-0-",
    "_logicnodeprotocoltype": "lock",
    "_logicnodetype": "lock",
    "_nodeid": 2,
    "_nodeprotocoltype": "lock",
    "_nodetype": "lock",
    "_properties": {
      "BatteryPercentage": 85,
      "Locked": true
    }
  }
}

```

4.1.6. 按扩展属性查询节点设备

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.GetNodeList",
  "params": {
    "WithoutProperties": <true/false>, //不获取设备状态属性, 缺省 false
    "Conditions": {"<attr>": "<value>"}, //属性:值, 查询条件可以指定多个(与的关系)
    "SortBy": ["<attr>"], //排序属性, attr 升序, -attr 降序, 排序可以指定多个
    "PageSize": <pagesize>, //每页记录数
    "Page": <page> //第几页
  }
}

```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功, 非 1-失败
  "errorinfo": "",         //错误信息
  "errorinfo_cn": "",      //本地化错误信息(Accept-Language 相关)
  "data": {
    "TotalCount": "<TotalCount>", //总记录数
    "PageCount": "<PageCount>",  //总页数
    "PageSize": "<pagesize>",    //每页记录数
    "Page": "<page>",            //当前第几页
    "PageList": [
      {
        "Name": "<value>",      //扩展属性名: 属性值
        ...
        "_deviceid": "<deviceid>", //内置的设备属性
        "_isentity": "<true/false>", //内置的设备属性
        "_nodeid": "<nodeid>", //内置的设备属性
        "_nodetype": "<devicetype>", //内置的设备属性
        "_nodeprotocoltype": "<nodeprotocoltype>", //内置的设备属性
        "_islogic": "<true/false>", //内置的设备属性
        "_logicid": "<logicid>", //内置的设备属性
        "_logicnodetype": "<logicnodetype>", //内置的设备属性
        "_logicnodeprotocoltype": "<logicnodeprotocoltype>", //内置的设备属性
        "_isbluetooth": true, //内置的设备属性
        "_bluetooth_mac": "<bluetooth MAC>", //内置的设备属性
        "_bluetooth_name": "<bluetooth name>", //内置的设备属性
        "_dsk": "", //内置的设备属性
        "_properties": { //内置的设备属性
          "Name": "<value>" //属性名: 属性值
          ...
        }
      }
    ]
  }
}
```

【注】

- 1) _打头的属性是系统内置的设备属性
- 2) 仅蓝牙设备有_isbluetooth、_bluetooth_mac、_bluetooth_name 这三个属性
- 3) 该接口不适合于以下使用方式：
业务应用系统周期性地全量同步设备及其状态

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'Content-Type: application/json' \
  -H 'Postman-Token: a4ebcb91-0832-4280-affe-49627626946a' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.GetNodeList",
    "params": {
      "Conditions": {"SSID": "test"},
      "SortBy": ["-Name"],
```

```
        "PageSize": 8,
        "Page": 1
    }
}'
//返回结果:
{
    "resultcode": 1,
    "data": {
        "TotalCount": 2,
        "PageCount": 1,
        "PageSize": 8,
        "Page": 1,
        "PageList": [
            {
                "Name": "test lock3",
                "SSID": "test",
                "_deviceid": "iSurpass2018100000",
                "_isentity": true,
                "_islogic": true,
                "_logicaid": "iSurpass2018100000-3-0-",
                "_logicnodeprotocoltype": "lock",
                "_logicnodetype": "lock",
                "_nodeid": 3,
                "_nodeprotocoltype": "lock",
                "_nodetype": "lock",
                "_properties": {
                    "BatteryPercentage": 85,
                    "Locked": true
                }
            },
            {
                "Name": "test lock2",
                "SSID": "test",
                "_deviceid": "iSurpass2018100000",
                "_isentity": true,
                "_islogic": true,
                "_logicaid": "iSurpass2018100000-2-0-",
                "_logicnodeprotocoltype": "lock",
                "_logicnodetype": "lock",
                "_nodeid": 2,
                "_nodeprotocoltype": "lock",
                "_nodetype": "lock",
                "_properties": {
                    "BatteryPercentage": 85,
                    "Locked": true
                }
            }
        ]
    }
}
```

4.1.7. 设置逻辑节点扩展属性

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.SetLogicNodeExtendedAttribute",
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "<Name>": <value> //扩展属性名: 属性值
  }
}
```

【注】属性名可以是任何字母打头的字符串, 属性值可以是任何 json 值(字符串, 整型, 浮点型, 布尔, 对象, 数组)

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCl6IjE5MzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'Content-Type: application/json' \
  -H 'Postman-Token: 406f8f51-5bed-476e-ac82-dc0688df3b98' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.SetLogicNodeExtendedAttribute",
    "logicnodeid": "iRemote7005000000020-21-3-",
    "params": {
      "Name": "开关 3",
      "Room": "餐厅"
    }
  }'
//返回结果:
{
  "resultcode": 1
}
```

4.1.8. 获取逻辑节点属性

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetLogicNodeAttribute",
  "logicnodeid": "<logicnodeid>", //逻辑节点 id
  "params": {
    "WithoutProperties": <true/false> //不获取设备状态属性, 缺省 false
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "Name": "<value>", //扩展属性名: 属性值
    ...
    "_deviceid": "<deviceid>", //内置的设备属性
    "_isentity": <true/false>, //内置的设备属性
    "_nodeid": "<nodeid>", //内置的设备属性
    "_nodetype": "<devicetype>", //内置的设备属性
    "_nodeprotocoltype": "<nodeprotocoltype>", //内置的设备属性
    "_islogic": <true/false>, //内置的设备属性
    "_logid": "<logid>", //内置的设备属性
    "_logicnodetype": "<devicetype>", //内置的设备属性
    "_logicnodeprotocoltype": "<logicnodeprotocoltype>", //内置的设备属性
    "_isbluetoothlock": true, //内置的设备属性
    "_bluetooth_mac": "<bluetooth MAC>", //内置的设备属性
    "_bluetooth_name": "<bluetooth name>", //内置的设备属性
    "_dsk": "", //内置的设备属性
    "_properties": { //内置的设备属性
      "Name": "<value>" //属性名: 属性值
      ...
    }
  }
}
```

【注】

- 1) _打头的属性是系统内置的设备属性
- 2) 仅蓝牙锁有_isbluetoothlock、_bluetooth_mac、_bluetooth_name 这三个属性

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'Content-Type: application/json' \
  -H 'Postman-Token: 043d0802-d7a1-4ff9-b7e4-eb005dcca176' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.GetLogicNodeAttribute",
    "logicnodeid": "iRemote7005000000020-21-3-"
  }'
```

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "Name": "开关 3",
    "Room": "餐厅",
    "_deviceid": "iRemote7005000000020",
    "_isentity": false,
    "_islogic": true,
    "_logicid": "iRemote7005000000020-21-3-",
    "_nodeprotocoltype": "switch",
    "_nodetype": "switch",
    "_nodeid": 21,
    "_logicnodeprotocoltype": "switch",
    "_logicnodetype": "switch",
    "_properties": {
      "IsFault": false,
      "Status": "off"
    }
  }
}
```

4.1.9. 按扩展属性查询逻辑节点

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
 Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.GetLogicNodeList",
  "params": {
    "WithoutProperties": <true/false>, //不获取设备状态属性, 缺省 false
    "Conditions": [{"<attr>":"<value>"}, //属性:值, 查询条件可以指定多个(与的关系)
    "SortBy": ["<attr>"], //排序属性, attr 升序, -attr 降序, 排序可以指定多个
    "PageSize": <pagesize>, //每页记录数
    "Page": <page> //第几页
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功, 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "TotalCount": <TotalCount> //总记录数
    "PageCount": <PageCount> //总页数
    "PageSize": <pagesize>, //每页记录数
    "Page": <page>, //当前第几页
    "PageList": [ //当前页的设备列表
      {
        "Name": "<value>", //扩展属性名: 属性值

```

```

...
    "_deviceid": "<deviceid>", //内置的设备属性
    "_isentity": <true/false>, //内置的设备属性
    "_nodeid": "<nodeid>", //内置的设备属性
    "_nodetype": "<devicetype>", //内置的设备属性
    "_nodeprotocoltype": "<nodeprotocoltype>", //内置的设备属性
    "_islogic": <true/false>, //内置的设备属性
    "_logicaid": "<logicaid>", //内置的设备属性
    "_logicnodetype": "<devicetype>", //内置的设备属性
    "_logicnodeprotocoltype": "<logicnodeprotocoltype>", //内置的设备属性
    "_isbluetooth": true, //内置的设备属性
    "_bluetooth_mac": "<bluetooth MAC>", //内置的设备属性
    "_bluetooth_name": "<bluetooth name>", //内置的设备属性
    "_dsk": "", //内置的设备属性
    "_properties": { //内置的设备属性
        "Name": "<value>" //属性名: 属性值
    }
}
...
]
}

```

- 1) _打头的属性是系统内置的设备属性
- 2) 仅蓝牙设备有_isbluetooth、_bluetooth_mac、_bluetooth_name 这三个属性
- 3) 该接口不适合于以下使用方式：
业务应用系统周期性地全量同步设备及其状态

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/app \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBBN-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'Content-Type: application/json' \
  -H 'Postman-Token: a4ebcb91-0832-4280-affe-49627626946a' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.GetLogicNodeList",
    "params": {
      "Conditions": {"Room": "餐厅"}
    }
  }'
//返回结果:
{
  "resultcode": 1,
  "data": {
    "TotalCount": 2,
    "PageCount": 1,
    "PageSize": 10,
    "Page": 1,
    "PageList": [
      {
        "_deviceid": "iRemote7005000000020",
        "_isentity": true,
        "_islogic": true,

```

```

        "_logiconodeprotocoltype": "lock",
        "_logiconodetype": "lock",
        "_nodeid": 124,
        "_nodeprotocoltype": "lock",
        "_nodetype": "lock",
        "_properties": {
            "BatteryPercentage": 99,
            "IsFault": false,
            "Locked": true
        }
    },
    {
        "Name": "开关 3",
        "Room": "餐厅",
        "_deviceid": "iRemote7005000000020",
        "_isentity": false,
        "_islogic": true,
        "_logiconodeprotocoltype": "switch",
        "_logiconodetype": "switch",
        "_nodeprotocoltype": "switch",
        "_nodetype": "switch",
        "_nodeid": 21,
        "_properties": {
            "IsFault": false,
            "Status": "off"
        }
    }
]
}
}

```

4.1.10. 添加设备到 app

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.BindApp",
    "deviceid": "<deviceid>",           //设备 ID
    "params": {
        "QID": "<qid>"                 //设备的 qid
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,           //1-成功 非 1-失败
}

```

```
"errorinfo": "",          //错误信息
"errorinfo_cn": ""        //本地化错误信息(Accept-Language 相关)
}
```

4.1.11. 设置应用属性

方法: post

请求头(header):

```
Accept-Language: <language>          //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.SetAppAttribute",
  "params": {
    "Key": "<key>",          //属性 key
    "Value": "<value>"      //属性值
  }
}
```

【注】属性名可以是任何字母打头的字符串

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": ""        //本地化错误信息(Accept-Language 相关)
}
```

4.1.12. 获取应用属性

方法: post

请求头(header):

```
Accept-Language: <language>          //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.GetAppAttribute",
  "params": {
    "Key": "<key>"          //属性 key
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
}
```

```

    "data": {
        "Key": "<key>",           //属性 key
        "Value": "<value>"       //属性值
    }
}

```

4.1.13. 注册应用

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.RegistApp",
    "params": {
        "AppKey": "<app key>",           //app key, 必须
        "AppSecret": "<app secret>",     //app secret, 必须
        "Name": "<name>",                 //应用名称, 可无
        "Description": "<description>",   //应用描述; 可无
        "Contacter": "<contacter>",      //联系人姓名, 可无
        "Phone": "<phone>",               //联系人电话; 可无
        "Weixin": "<weixin>",            //联系人微信, 可无
        "Email": "<email>",               //联系人 E-Mail, 可无
        "AppHome": "<home url>"          //应用首页 URL, 可无
    }
}

```

【注】仅限当前 appkey 为 admin 可调用此接口

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,           //1-成功 非 1-失败
    "errorinfo": "",           //错误信息
    "errorinfo_cn": ""         //本地化错误信息(Accept-Language 相关)
}

```

4.1.14. 更改应用注册信息

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.UpdateApp",
    "params": {
        "AppKey": "<app key>",           //app key, 当前 appkey 为 admin 必须指定目标应用的 appkey
    }
}

```

```

        "Name": "<name>", //应用名称, 可无
        "Description": "<description>", //应用描述; 可无
        "Contacter": "<contacter>", //联系人姓名, 可无
        "Phone": "<phone>", //联系人电话; 可无
        "Weixin": "<weixin>", //联系人微信, 可无
        "Email": "<email>", //联系人 E-Mail, 可无
        "AppHome": "<home url>" //应用首页 URL, 可无
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

4.1.15. 注销应用

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.UnRegistApp",
    "params": {
        "AppKey": "<app key>" //app key, 必须
    }
}

```

【注】仅限当前 appkey 为 admin 可调用此接口

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

4.1.16. 注册设备

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.RegistDevice",

```

```

"params": {
  "DeviceList": [           //设备列表, 必须
    {
      "type": "<type>",      //设备类型, gateway:网关
      "protocoltype": "<type>", //设备数据协议类型, 可无(缺省为设备类型)
      "ptype": "<type>",     //设备通讯协议类型, 可无
      "deviceid": "<Device ID>", //设备 ID
      "qid": "<QRCode>",      //二维码
      "qtype": "<type>",      //二维码类型, 可无(缺省为设备类型)
      "secret": "<secret>",    //设备密钥
      "isbluetooth": "<true/false>", //是否是蓝牙设备
      "bluetooth_name": "<name>", //蓝牙名称
      "bluetooth_mac": "<mac>", //蓝牙 MAC
      "bluetooth_secret": "<secret>", //蓝牙设备密钥
      "bluetooth_user_secret0": "<secret>", //蓝牙用户密钥
      "bluetooth_data_secret0": "<secret>" //蓝牙通讯密钥
    }
  ]
}

```

【注】仅限当前 appkey 为 admin 可调用此接口

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,          //1-成功 非 1-失败
  "errorinfo": ""          //错误信息
}

```

4.2. 锁高级服务

*锁相关数据存储依赖于 MongoDB

url: http(s)://<ip>:<port>/app/lock

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

4.2.1. 获取锁的用户列表

*仅限于通过开放服务接口远程创建的锁用户（不包括在锁上手工添加的锁用户）。

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetLockUserList",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
  "params": {
    "Conditions": {"<attr>":"<value>"}, //属性:值, 查询条件可以指定多个(与的关系)
    "SortBy": ["<attr>"] //排序属性, attr 升序, -attr 降序, 排序可以指定多个
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功, 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "LockUserList": [
      {
        "UserType": "<UserType>", //1-密码 2-指纹 3-MF 卡 4-身份证
        //9-蓝牙/二维码 90-蓝牙钥匙扣 7-人脸
        "UserLevel": "<UserLevel>", //1-管理员 2-普通用户 4-一次性临时密码(UserType 必须为 1)
        "UserID": "<UserID>", //UserID 规则:
        //管理员: 1~9
        //普通用户: 10~230
        //一次性临时密码: 243~252
        "UserPassword": "<UserPassword>", //密码串或指纹数据串或卡数据
        //卡数据: MF 卡, 4byte 值转换为十六进制串
        //身份证卡, 8byte 值转换为十六进制串
        "ValidWeekDays": "<SumWeekdayValue>", //累加值: 如 5 表示星期天和星期二有效
        //1-星期天 2-星期一 4-星期二 8-星期三
        //16-星期四 32-星期五 64-星期六
        "ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间
        "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
        "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
        "UserName": "<user name>", //用户名称
        "Owner": "<owner account>", //owner 用户
        "Paused": "<true/false>" //停用状态(true/false)
      }
    ]
  }
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.GetLockUserList",
    "deviceid": "iSurpass2019100001",
    "nodeid": 2
  }'
```

```
//返回结果:
{
  "resultcode": 1,
  "data": {
    "LockUserList": [
      {
        "UserID": 10,
        "UserLevel": 2,
        "UserPassword": "1111",
        "UserType": 1,
        "ValidBeginDateTime": "2019-04-10 09:00",
        "ValidEndDateTime": "2019-04-10 23:59",
        "ValidWeekDays": 127,
        "ValidWeekTime": "00:00~23:59",
        "_deviceid": "iSurpass2019100001",
        "_nodeid": 2
      },
      {
        "UserID": 31,
        "UserLevel": 2,
        "UserPassword": "96AF6269",
        "UserType": 3,
        "ValidBeginDateTime": "2019-04-01 09:00",
        "ValidEndDateTime": "2019-04-30 23:59",
        "ValidWeekDays": 127,
        "ValidWeekTime": "00:00~23:59",
        "_deviceid": "iSurpass2019100001",
        "_nodeid": 2
      }
    ]
  }
}
```

4.2.2. 更改指定类型锁用户的有效日期时间

*仅限于通过开放服务接口远程创建的锁用户（不包括在锁上手工添加的锁用户），锁用户的数据存储依赖于 MongoDB。

方法：post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.UpdateLockUserValidDateTime",
  "async": false,                      //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>",             //网关设备 ID
  "nodeid": "<nodeid>",                 //智能锁在网关设备上的节点 id
  "params": {
    "UserType": "<UserType>",           //0-全部 1-密码 2-指纹 3-MF 卡 4-身份证
    "UserLevel": "<UserLevel>",         //0-全部 1-管理员 2-普通用户 4-一次性临时密码
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
  }
}
```

```

        "ValidEndDateTime": "<YYYY-MM-DD HH:MM>" //有效期的终止日期时间
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "receipt": "<receipt>", //异步调用的回执 id
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '
{
  "method": "thing.service.UpdateLockUserValidDateTime",
  "async": true,
  "deviceid": "iSurpass2019100001",
  "nodeid": 2,
  "params": {
    "UserType": 3,
    "UserLevel": 0,
    "ValidBeginDateTime": "2019-04-10 00:00",
    "ValidEndDateTime": "2019-04-30 23:59"
  }
}
'
//返回结果:
{
  "resultcode": 1,
  "receipt": "lock-1554945755303"
}

```

4.2.3. 添加/修改密码/临时密码用户

*锁用户的数据存储依赖于 MongoDB。

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.AddPasswordUser",

```

```

"async": false, //false-同步模式(缺省), true-异步模式
"deviceid": "<deviceid>", //网关设备 ID
"nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
"params": {
  "UserLevel": "<UserLevel>", //1-管理员 2-普通用户 4-一次性临时密码(UserType 必须为 1)
  "UserID": "<UserID>", //UserID 规则:
    系统分配: 0
    管理员: 1~9
    普通用户: 10~230
    一次性临时密码: 243~252
  "UserPassword": "<UserPassword>", //密码
  "ValidWeekDays": "<SumWeekdayValue>", //有效星期累加值: 如 5 表示星期天和星期二有效
    1-星期天 2-星期一 4-星期二 8-星期三
    16-星期四 32-星期五 64-星期六
  "ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间
  "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
  "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
  "UserName": "<UserName>" //用户名
}
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功, 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "UserID": "<UserID>" //锁用户 ID
  }
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.AddPasswordUser",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params": {
      "UserLevel": 1,
      "UserID": 0,
      "UserPassword": "8888",
      "UserName": "d"
    }
  }
'

```

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "UserID": 5
  }
}
```

4.2.4. 设置锁用户名称

*锁用户的数据存储依赖于 MongoDB。

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.SetUserName",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": <nodeid>,                 //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID>,               //UserID
    "UserName": "<UserName>"         //用户名
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,           //1-成功, 非 1-失败
  "errorinfo": "",           //错误信息
  "errorinfo_cn": ""        //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.SetUserName",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params": {
      "UserID": 5,
      "UserName": "admin"
    }
  }
,'
```

//返回结果:

```
{
```

```
"resultcode": 1
}
```

4.2.5. 添加卡

*锁用户的数据存储依赖于 MongoDB。

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.AddCard",
  "async": false,                        //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid": "<nodeid>",                  //智能锁在网关设备上的节点 id
  "params": {
    "UserName": "<UserName>",           //用户名
    "UserID": "<UserID>",               //锁用户 ID: 1~230 (未指定则由锁自动分配)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                      //1-成功 非 1-失败
  "receipt": "<receipt>",               //异步调用的回执 id
  "errorinfo": "",                      //错误信息
  "errorinfo_cn": "",                  //本地化错误信息(Accept-Language 相关)
  "params": {
    "UserType": "<UserType>",           //1-密码 2-指纹 3-MF 卡 4-身份证
    "UserLevel": "<UserLevel>",         //1-管理员 2-普通用户 4-一次性临时密码(UserType 必须为 1)
    "UserID": "<UserID>",               //卡用户 ID
    "UserPassword": "<UserPassword>"   //卡数据
  }
}
```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.AddCard",
```

```

        "deviceid": "iSurpass2018100000",
        "nodeid": 2,
        "params": {
            "UserName": "cardx"
        }
    }
}
//返回结果:
{
    "params": {
        "UserID": 20,
        "UserLevel": 2,
        "UserPassword": "C7AF0BD6",
        "UserType": 3
    },
    "resultcode": 1
}

```

4.2.6. 添加指纹

*锁用户的数据存储依赖于 MongoDB。

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.AddFingerPrint",
    "async": false,                        //false-同步模式(缺省), true-异步模式
    "deviceid": "<deviceid>",             //网关设备 ID
    "nodeid": <nodeid>,                   //智能锁在网关设备上的节点 id
    "params": {
        "UserName": "<UserName>",         //用户名
        "UserID": <UserID>                //锁用户 ID: 1~230 (未指定则由锁自动分配)
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,                      //1-成功 非 1-失败
    "receipt": "<receipt>",               //异步调用的回执 id
    "errorinfo": "",                      //错误信息
    "errorinfo_cn": "",                  //本地化错误信息(Accept-Language 相关)
    "params": {
        "UserType": <UserType>,          //1-密码 2-指纹 3-MF 卡 4-身份证
        "UserLevel": <UserLevel>,        //1-管理员 2-普通用户 4-一次性临时密码(UserType 必须为 1)
        "UserID": <UserID>                //指纹用户 ID
    }
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcGlzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.AddFingerPrint",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params": {
      "UserName": "my fingerprint"
    }
  }'
//返回结果:
{
  "params": {
    "UserID": 33,
    "UserLevel": 2,
    "UserType": 2
  },
  "resultcode": 1
}
```

4.2.7. 添加/修改锁用户

*锁用户的数据存储依赖于 MongoDB。

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.AddUser",
  "async": false, //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
  "params": {
    "UserType": <UserType>, //1-密码 2-指纹 3-MF 卡 4-身份证 7-人脸
    "UserLevel": <UserLevel>, //1-管理员 2-普通用户 4-一次性临时密码(UserType 必须为 1)
    "UserID": <UserID>, //UserID 规则:
    //系统分配: 0
    //管理员: 1~9
    //普通用户: 10~230
    //一次性临时密码: 243~252
    "UserPassword": "<UserPassword>", //密码串或指纹数据串或卡数据
    //卡数据: MF 卡, 4byte 值转换为十六进制串
    //身份证卡, 8byte 值转换为十六进制串
  }
}
```

```

"FaceFileURL": "<URL path>", //人脸照片文件的 URL path
"ValidWeekDays": "<SumWeekdayValue>", //有效星期累加值: 如 5 表示星期天和星期二有效
                                     1-星期天 2-星期一 4-星期二 8-星期三
                                     16-星期四 32-星期五 64-星期六
"ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间, 多个时间段,分隔
"ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
"ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
"UserName": "<UserName>", //用户名
"CustomInfo": "<HEX>" //定制数据(HEX 字符串)
}
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功, 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "UserID": "<UserID>" //锁用户 ID
  }
}

```

【注】

- 1) 当 async 参数为 true 时, resultcode 为 1 仅表示服务器接收了请求、不表示操作成功, 通常发异步请求前你的系统需先建立 websocket 来接收操作结果, 参见[接收异步请求的操作结果](#)
- 2) NB-IoT 锁、WIFI 锁 async 参数值必须为 true, nodeid 参数值必须为 1
- 3) 目前仅 protocoltype 为 entranceb 的设备支持 CustomInfo: 64 个 HEX 字符
- 4) 仅部分设备 ValidWeekTime 支持多个时间段

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.AddUser",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params": {
      "UserType": 1,
      "UserLevel": 2,
      "UserID": 0,
      "UserPassword": "5678",
      "ValidBeginDateTime": "2019-08-30 00:00",
      "ValidEndDateTime": "2018-12-31 23:59",
      "ValidWeekDays": 127,
      "ValidWeekTime": "00:00~23:59",
      "UserName": "u5678"
    }
  }'
//返回结果:
{
  "resultcode": 1,

```

```

    "data": {
      "UserID": 175
    }
  }
}

```

4.2.8. 查询锁异步请求历史

*锁异步请求记录依赖于 MongoDB 存储。

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.QueryLockAsyncInvoke",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": "<nodeid>",               //智能锁在网关设备上的节点 id
  "params": {
    "Conditions": {"<attr>":"<value>"}, //属性:值, 查询条件可以指定多个(与的关系)
    "SortBy": ["<attr>"],             //排序属性, attr 升序, -attr 降序, 排序可以指定多个
    "PageSize": "<pagesize>",         //每页记录数
    "Page": "<page>"                  //第几页
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,           //1-成功, 非 1-失败
  "errorinfo": "",           //错误信息
  "errorinfo_cn": "",        //本地化错误信息(Accept-Language 相关)
  "data": {
    "TotalCount": "<TotalCount>"    //总记录数
    "PageCount": "<PageCount>"      //总页数
    "PageSize": "<pagesize>",        //每页记录数
    "Page": "<page>",                //当前第几页
    "PageList": [                //当前页的设备列表
      {
        "_deviceid": "<deviceid>",    //网关设备 ID
        "_nodeid": "<nodeid>",        //智能锁在网关设备上的节点 id
        "_method": "<method name>",  //异步请求方法名
        "_params": {                //异步请求参数
          ...
        },
        "_processresult": "<resultcode>", //异步请求结果: 0-待下发, 1-成功, -9-取消, 其它值失败
        "_receipt": "<receipt>",      //异步请求的回执 id
      }
      ...
    ]
  }
}

```

```
}
```

[示例 1]查询待下发的创建锁用户的异步请求

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'Content-Type: application/json' \
  -H 'Postman-Token: a4ebcb91-0832-4280-affe-49627626946a' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.QueryLockAsyncInvoke",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params": {
      "Conditions": {"_method": "thing.service.AddUser", "_processresult": 0},
      "SortBy": ["_receipt"],
      "PageSize": 10,
      "Page": 1
    }
  }'
```

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "TotalCount": 2,
    "PageCount": 1,
    "PageSize": 10,
    "Page": 1,
    "PageList": [
      {
        "_deviceid": "iSurpass2018100000",
        "_method": "thing.service.AddUser",
        "_nodeid": 2,
        "_params": {
          "UserID": 250,
          "UserLevel": 4,
          "UserName": "t4444",
          "UserPassword": "4444",
          "UserType": 1
        },
        "_processresult": 0,
        "_receipt": "lock-1567067828105"
      },
      {
        "_deviceid": "iSurpass2018100000",
        "_method": "thing.service.AddUser",
        "_nodeid": 2,
        "_params": {
          "UserID": 4,
          "UserLevel": 1,
          "UserName": "admin",
          "UserPassword": "111111",
          "UserType": 1
        }
      }
    ]
  }
}
```

```

        "_processresult": 0,
        "_receipt": "lock-1567067842440"
    }
}
]
}
}
}

```

[示例 2]查询操作失败的创建锁用户的异步请求

```

curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'Content-Type: application/json' \
  -H 'Postman-Token: a4ebcb91-0832-4280-affe-49627626946a' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.QueryLockAsyncInvoke",
    "deviceid": "iSurpass2018100000",
    "nodeid": 2,
    "params": {
      "Conditions": {
        "_method": "thing.service.AddUser",
        "_processresult": {"$nin": [0,1]}
      },
      "SortBy": ["-_receipt"],
      "PageSize": 10,
      "Page": 1
    }
  }'

```

//返回结果:

```

{
  "resultcode": 1,
  "data": {
    "TotalCount": 1,
    "PageCount": 1,
    "PageSize": 10,
    "Page": 1,
    "PageList": [
      {
        "_deviceid": "iSurpass2018100000",
        "_method": "thing.service.AddUser",
        "_nodeid": 2,
        "_params": {
          "UserID": 55,
          "UserLevel": 2,
          "UserName": "card a",
          "UserPassword": "AFAFAFAF",
          "UserType": 3,
          "ValidBeginDateTime": "2019-08-30 00:00",
          "ValidEndDateTime": "2018-12-31 23:59",
          "ValidWeekDays": 127,
          "ValidWeekTime": "00:00~23:59"
        },
        "_processresult": 4,
        "_icode": "ucode_lock_8003_3",

```

```

        "_processresult_cn": "Password already exists",
        "_receipt": "lock-1567145449326"
    }
}
]
}
}

```

4.2.9. 添加/修改蓝牙/二维码用户

*锁用户的数据存储依赖于 MongoDB。

方法：post

请求头(header)：

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数：

```

{
  "method": "thing.service.AddBluetoothUser",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": "<nodeid>",               //智能锁在网关设备上的节点 id
  "params": {
    "UserLevel": "<UserLevel>",      //1-管理员 2-普通用户
    "UserID": "<UserID>",             //UserID 规则:
                                     系统分配: 0
                                     管理员: 1~9
                                     普通用户: 10~230
    "ValidWeekDays": "<SumWeekdayValue>", //有效星期累加值: 如 5 表示星期天和星期二有效
                                     1-星期天 2-星期一 4-星期二 8-星期三
                                     16-星期四 32-星期五 64-星期六
    "ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
    "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
    "UserPassword": "<UserPassword>", //蓝牙用户验证密码(系统侧)
    "UserName": "<user name>",        //用户名称
    "Owner": "<owner account>"        //owner
  }
}

```

响应结果(Content-Type: application/json)：

```

{
  "resultcode": 1,           //1-成功, 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
  "data": {
    "UserID": "<UserID>"        //锁用户 ID
  }
}

```

【注】

1) 蓝牙锁/蓝牙+NB-IoT 锁, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.AddBluetoothUser",
    "deviceId": "iRemote2005100006772",
    "nodeid": 1,
    "params": {
      "UserLevel": 1,
      "UserID": 2,
      "UserPassword": "222222",
      "UserName": "bluetooth2",
      "Owner": "123838100@qq.com"
    }
  }'
//返回结果:
{
  "resultcode": 1,
  "data": {
    "UserID": 2
  }
}
```

4.2.10. 修改蓝牙/二维码用户的有效日期时间

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.UpdateBluetoothUserValidDateTime",
  "deviceId": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID>, //蓝牙用户 ID
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
    "ValidEndDateTime": "<YYYY-MM-DD HH:MM>" //有效期的终止日期时间
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 蓝牙锁/蓝牙+NB-IoT 锁, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d {
    "method": "thing.service.UpdateBluetoothUserValidDateTime",
    "deviceid": "iRemote2005100006772",
    "nodeid": 1,
    "params": {
      "UserID": 2,
      "ValidBeginDateTime": "2019-09-25 00:00",
      "ValidEndDateTime": "2019-09-30 23:59"
    }
  }
//返回结果:
{
  "resultcode": 1
}
```

4.2.11. 修改蓝牙/二维码用户的有效星期

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.UpdateBluetoothUserValidWeekTime",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID>, //蓝牙用户 ID
    "ValidWeekDays": <SumWeekdayValue>, //有效星期累加值: 如 5 表示星期天和星期二有效
    //1-星期天 2-星期一 4-星期二 8-星期三
    //16-星期四 32-星期五 64-星期六
    "ValidWeekTime": "<HH:MM~HH:MM>" //有效星期的起止时间
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "receipt": "<receipt>", //异步调用的回执 id
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 蓝牙锁/蓝牙+NB-IoT 锁, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBBN-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d {
    "method": "thing.service.UpdateBluetoothUserValidWeekTime",
    "deviceid": "iRemote2005100006772",
    "nodeid": 1,
    "params":{
      "UserID": 2,
      "ValidWeekDays": 96,
      "ValidWeekTime": "00:00~23:59"
    }
  }
//返回结果:
{
  "resultcode": 1
}
```

4.2.12. 删除蓝牙/二维码用户

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.DeleteBluetoothUser",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID> //蓝牙用户 ID
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 蓝牙锁/蓝牙+NB-IoT 锁, nodeid 参数值必须为 1

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.DeleteBluetoothUser",
    "deviceid": "iRemote2005100006772",
    "nodeid": 1,
    "params": {
      "UserID": 2
    }
  }'
//返回结果:
{
  "resultcode": 1
}

```

4.2.13. 获取蓝牙/二维码用户钥匙

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.GetBluetoothKey",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": <nodeid>,                 //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID>,               //蓝牙用户 ID
    "UserPassword": "<UserPassword>" //蓝牙用户验证密码
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,           //1-成功 非 1-失败
  "errorinfo": "",           //错误信息
  "errorinfo_cn": "",        //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothUserKey": "<bluetooth user key>", //蓝牙钥匙, hex 串
    "BluetoothDataSecret": "<bluetooth data secret>", //蓝牙通讯加密密钥, hex 串
    "UnixTimestamp": <unix timestamp>, //系统当前时间(unix 时间戳)
    "SuperUserKey": "<bluetooth super user key>", //超级用户蓝牙钥匙, hex 串
    "SuperDataSecret": "<bluetooth super user data secret>", //超级用户蓝牙通讯加密密钥, hex 串
    "BluetoothPassword": "<bluetooth password>" //蓝牙开锁密码, hex 串
  }
}

```

【注】

- 1) 蓝牙锁/蓝牙+NB-IoT 锁, nodeid 参数值必须为 1
- 2) BluetoothUserKey 将用于手机蓝牙开锁, BluetoothDataSecret 将用于手机与蓝牙锁通讯加密
- 3) BluetoothPassword 适用于蓝牙手机透传数据给蓝牙锁(手机与蓝牙锁之间通讯不做加密)

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzcy0FwcGlzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.GetBluetoothKey",
    "deviceid": "iRemote2005100009667",
    "nodeid": 1,
    "params": {
      "UserID": 0,
      "UserPassword": "123456"
    }
  }'
//返回结果:
{
  "resultcode": 1,
  "data": {
    "BluetoothDataSecret": "114F10C9BC0A36813331C7490AE46360",
    "BluetoothPassword": "F059CBD19307805A1E1CE6DBA7E4AAFA79CE30E9",
    "BluetoothUserKey": "1E06FB45430FFDE103322E5DE09691BC6F96FA88B2CFF547A70B138EA3025C30",
    "SuperDataSecret": "114F10C9BC0A36813331C7490AE46360",
    "SuperUserKey": "5D9315DDF40E5D6EAD7979B840D9067CCFCB678AF846BFD8BB94A259B5DB1CA",
    "UnixTimestamp": 1577776634
  }
}
```

4.2.14. 更换蓝牙密钥

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.UpdateBluetoothSecrets",
  "deviceid": "<deviceid>",              //网关设备 ID
  "nodeid": <nodeid>,                   //智能锁在网关设备上的节点 id
  "params": {
    "UserID": <UserID>,                 //(管理员)蓝牙用户 ID
    "UserPassword": "<UserPassword>",   //蓝牙用户验证密码
    "BluetoothSecrets": "<bluetooth secret data>" // (锁生成)蓝牙密钥 TEA 加密数据, hex 串
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
  "errorinfo": "",      //错误信息
  "errorinfo_cn": "",    //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothUserKey": "<bluetooth user key>",    //(管理员)新的蓝牙钥匙,
    "BluetoothDataSecret": "<bluetooth data secret>", //新的蓝牙通讯加密密钥
    "UnixTimestamp": "<unix timestamp>",          //系统当前时间(unix 时间戳)
    "SuperUserKey": "<bluetooth super user key>",    //超级用户蓝牙钥匙, hex 串
    "SuperDataSecret": "<bluetooth super user data secret>" //超级用户蓝牙通讯加密密钥, hex 串
  }
}
```

【注】

- 1) 蓝牙锁/蓝牙+NB-IoT 锁, nodeid 参数值必须为 1
- 2) BluetoothUserKey 将用于请求蓝牙锁确认更换蓝牙密钥, BluetoothDataSecret 将用于手机与蓝牙锁通讯加密

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGZlc0FwcGlzInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.UpdateBluetoothSecrets",
    "deviceid": "iRemote2005100006772",
    "nodeid": 1,
    "params": {
      "UserID": 2,
      "UserPassword": "222222",
      "BluetoothSecrets":
        "B4676C9A7162EDBCF614826D5105A5EBD7AEAC201ECF3C03BE4E383560BB45D2A8E907EDAC38BAD45FA14393D6C230F8"
    }
  }'
```

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "BluetoothDataSecret": "5262F34559BD59C52E48A71835C959A4",
    "BluetoothUserKey": "337D9F7E84C3963DCD181DEC8201CF77BF2425D3410463C06E924167D89D7CE3",
    "SuperDataSecret": "5262F34559BD59C52E48A71835C959A4",
    "SuperUserKey": "99E75C240D803C66C971460DFEF213F190D983E2711844560E41CA28D2AD70A7",
    "UnixTimestamp": 1572316063,
    "UserName": "bluetooth2"
  }
}
```

4.2.15. 复位蓝牙密钥

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.ResetBluetoothSecrets",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
  "params": {
    "UserID": "<UserID>", //蓝牙用户 ID(管理员)
    "UserPassword": "<UserPassword>" //蓝牙用户验证密码
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

1) 蓝牙锁/蓝牙+NB-IoT 锁, nodeid 参数值必须为 1

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCl6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d {
    "method": "thing.service.ResetBluetoothSecrets",
    "deviceid": "iRemote2005100006772",
    "nodeid": 1,
    "params": {
      "UserID": 2,
      "UserPassword": "222222"
    }
  }
```

//返回结果:

```
{
  "resultcode": 1
}
```

【注】

蓝牙密钥恢复成出厂时的缺省值

4.2.16. 按 Owner 获取锁用户列表

*仅限于通过开放服务接口远程创建的锁用户（不包括在锁上手工添加的锁用户）。

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.GetLockUserListByOwner",
  "params": {
    "Owner": "<owner>" //owner
    "Conditions": {"<attr>":"<value>"}, //属性:值, 查询条件可以指定多个(与的关系)
    "SortBy": ["<attr>"] //排序属性, attr 升序, -attr 降序, 排序可以指定多个
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功, 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "LockUserList": [
      {
        "UserType": <UserType>, //1-密码 2-指纹 3-MF 卡 4-身份证 9-蓝牙
        "UserLevel": <UserLevel>, //1-管理员 2-普通用户 4-一次性临时密码(UserType 必须为 1)
        "UserID": <UserID>, //UserID 规则:
          管理员: 1~9
          普通用户: 10~230
          一次性临时密码: 243~252
        "UserPassword": "<UserPassword>", //密码串或指纹数据串或卡数据
          卡数据: MF 卡, 4byte 值转换为十六进制串
          身份证卡, 8byte 值转换为十六进制串
        "ValidWeekDays": <SumWeekdayValue>, //累加值: 如 5 表示星期天和星期二有效
          1-星期天 2-星期一 4-星期二 8-星期三
          16-星期四 32-星期五 64-星期六
        "ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间
        "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
        "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
        "UserName": "<user name>", //用户名称
        "Owner": "<owner account>", //owner 用户
        "Lock": {
          "Name": "<value>", //扩展属性名: 属性值
          ...
          "_deviceid": "<deviceid>", //内置的设备属性
          "_isentity": <true/false>, //内置的设备属性
          "_nodeid": <nodeid>, //内置的设备属性
          "_nodetype": "<devicetype>", //内置的设备属性
          "_nodeprotocoltype": "<nodeprotocoltype>", //内置的设备属性
        }
      }
    ]
  }
}
```

```

        "_islogic": "<true/false>", //内置的设备属性
        "_logid": "<logid>", //内置的设备属性
        "_logicnodetype": "<logicnodetype>", //内置的设备属性
        "_logicnodeprotocoltype": "<logicnodeprotocoltype>", //内置的设备属性
        "_isbluetoothlock": true, //内置的设备属性
        "_bluetooth_mac": "<bluetooth MAC>", //内置的设备属性
        "_bluetooth_name": "<bluetooth name>", //内置的设备属性
        "_properties": { //内置的设备属性
            "Name": "<value>" //属性名: 属性值
            ...
        }
    }
    ...
}
]
}
}

```

[示例]

```

curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d {
    "method": "thing.service.GetLockUserListByOwner",
    "params": {
      "Owner": "123838100@qq.com",
      "Conditions": {"UserType": 9}
    }
  }

```

//返回结果:

```

{
  "resultcode": 1,
  "data": {
    "LockUserList": [
      {
        "Lock": {
          "Name": "蓝牙锁 iRemote2005100006772",
          "_bluetooth_mac": "DC:2C:26:D2:79:1B",
          "_bluetooth_name": "bt2005100006772",
          "_deviceid": "iRemote2005100006772",
          "_isbluetoothlock": true,
          "_isentity": true,
          "_islogic": true,
          "_logid": "iRemote2005100006772-1-0-",
          "_logicnodetype": "lock",
          "_nodeid": 1,
          "_nodetype": "lock"
        },
        "Owner": "123838100@qq.com",
        "UserID": 2,
        "UserLevel": 1,
        "UserName": "bluetooth2",
        "UserPassword": "222222",

```

```

        "UserType": 9,
        "_deviceid": "iRemote2005100006772",
        "_nodeid": 1
    }
]
}
}

```

4.2.17. 异步批量添加/修改锁用户

*锁用户的数据存储依赖于 MongoDB。

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.AsyncBatchAddUsers",
    "deviceid": "<deviceid>",           //网关设备 ID
    "nodeid": <nodeid>,                 //智能锁在网关设备上的节点 id
    "params": {
        "UserList": [                 //锁用户数组
            {
                "UserType": <UserType>, //1-密码 2-指纹 3-MF 卡 4-身份证
                "UserLevel": <UserLevel>, //1-管理员 2-普通用户 4-一次性临时密码(UserType 必须为 1)
                "UserID": <UserID>,       //UserID 规则:
                    系统分配:           0
                    管理员:             1~9
                    普通用户:           10~230
                    一次性临时密码:     243~252
                "UserPassword": "<UserPassword>", //密码串或指纹数据串或卡数据
                    卡数据: MF 卡, 4byte 值转换为十六进制串
                    身份证卡, 8byte 值转换为十六进制串
                "ValidWeekDays": <SumWeekdayValue>, //有效星期累加值: 如 5 表示星期天和星期二有效
                    1-星期天 2-星期一 4-星期二 8-星期三
                    16-星期四 32-星期五 64-星期六
                "ValidWeekTime": "<HH:MM~HH:MM>", //有效星期的起止时间
                "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
                "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
                "UserName": "<UserName>"           //用户名
            }
        ]
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,           //1-成功, 非 1-失败
    "receipt": "<receipt>",    //异步调用的回执 id
    "errorinfo": "",          //错误信息
    "errorinfo_cn": ""        //本地化错误信息(Accept-Language 相关)
}

```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/lock \
  -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslhRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -H 'cache-control: no-cache' \
  -d '{
    "method": "thing.service.AsyncBatchAddUsers",
    "deviceid": "iSurpass2018000000",
    "nodeid": 2,
    "params": {
      "UserList": [
        {
          "UserType": 1,
          "UserLevel": 2,
          "UserID": 0,
          "UserPassword": "3333",
          "ValidBeginDateTime": "2019-08-30 00:00",
          "ValidEndDateTime": "2019-12-31 23:59",
          "UserName": "3"
        },
        {
          "UserType": 1,
          "UserLevel": 2,
          "UserID": 0,
          "UserPassword": "4444",
          "ValidBeginDateTime": "2019-08-30 00:00",
          "ValidEndDateTime": "2019-12-31 23:59",
          "UserName": "4"
        }
      ]
    }
  }'
//返回结果:
{
  "resultcode": 1,
  "receipt": "lock-1576551618563"
}
```

4.2.18. 异步批量删除锁用户

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.AsyncBatchDeleteUsers",
  "deviceid": "<deviceid>", //网关设备 ID
}
```

```

    "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
    "params": {
        "Conditions": {"<attr>":"<value>"} //属性:值, 查询条件可以指定多个(与的关系)
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "receipt": "<receipt>", //异步调用的回执 id
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

[示例]删除全部无限期的密码用户

```

curl -X POST \
    http(s)://112.74.20.113:8888/app/lock \
    -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
    -H 'cache-control: no-cache' \
    -d '{
        "method": "thing.service.AsyncBatchDeleteUsers",
        "deviceid": "iSurpass2018000000",
        "nodeid": 2,
        "params": {
            "Conditions": {"UserType": 1, "ValidBeginDateTime": {"$exists": false}}
        }
    }'
//返回结果:
{
    "resultcode": 1,
    "receipt": "lock-1576562047793"
}

```

4.2.19. 蓝牙加卡

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.BluetoothAddCard",
    "deviceid": "<deviceid>", //网关设备 ID
    "nodeid": <nodeid> //智能锁在网关设备上的节点 id
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
}

```

```

"errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
"data": {
  "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
  "BluetoothName": "<bluetooth name>", //蓝牙名称
  "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
}
}

```

4.2.20. 获取酒店锁开锁密码

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.GetHotelLockPassword",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
  "params": {
    "StartDate": "<YYYY-MM-DD HH>", //开始日期时间(精确到小时)
    "ValidHours": "<valid hours>" //有效时长(小时): 0~63, 0 表示不限时(超级开锁密码)
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "Password": "<password>" //开锁密码(8 位数字)
  }
}

```

【注】

NB-IoT 锁、WIFI 锁 nodeid 参数值必须为 1

当 ValidHours 参数值为 0 时, StartDate 参数值必须等于或大于上一次调用时的值

[示例]

curl -X POST \

http(s)://112.74.20.113:8888/app/lock \

-H 'Authorization: Bearer

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \

-H 'cache-control: no-cache' \

-d '{

"method": "thing.service.GetHotelLockPassword",

"deviceid": "iSurpass2018100000",

"nodeid": 2,

"params": {

"StartDate": "2020-08-03 16",

"ValidHours": 24

```

    }
  }
  //返回结果:
  {
    "data": {
      "Password ": "60231937"
    },
    "resultcode": 1
  }

```

4.2.21. 蓝牙加密码用户

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.BluetoothAddPassword",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": "<nodeid>",               //智能锁在网关设备上的节点 id
  "params": {
    "UserPassword": "<UserPassword>", //密码(4~6 位数字)
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
    "ValidEndDateTime": "<YYYY-MM-DD HH:MM>", //有效期的终止日期时间
    "UserID": "<UserID>"              //锁用户 ID: 1~230 (未指定或 255 则由锁自动分配)
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,           //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothAddCardCmd": "<data>"      //蓝牙指令数据, hex 串
  }
}

```

【注】

1) 同一把锁同 1 秒内只能获取 1 次(同 1 秒内多次调用返回结果相同);

若是纯蓝牙锁(或是 NB-IoT 网络不畅通的情形下), 为了让锁的系统时钟正确, 建议 App 与锁蓝牙通讯时调用: [蓝牙将服务器的系统时间同步到锁](#)

2) 指定 UserID(需 openservice v2.8.0 及以上版本)时不能同时指定 ValidBeginDateTime 和 ValidEndDateTime, 有效期设定参见[蓝牙设置锁用户的有效日期时间](#)

4.2.22. 蓝牙将服务器的系统时间同步到锁

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothSyncTime",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid> //智能锁在网关设备上的节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
  }
}
```

【注】

同一把锁同 1 秒内只能获取 1 次(同 1 秒内多次调用返回结果相同);

若是纯蓝牙锁(或是 NB-IoT 网络不畅通的情形下), 为了让锁的系统时钟正确, 建议 App 与锁蓝牙通讯时调用: [蓝牙将服务器的系统时间同步到锁](#)

4.2.23. 添加蓝牙钥匙扣用户

*锁用户的数据存储依赖于 MongoDB。

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.AddBluetoothKeyRingUser",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
  "params": {
    "UserName": "<UserName>" //用户名
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
```

```

"errorinfo": "",          //错误信息
"errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
"data": {
  "BluetoothMAC": "<bluetooth MAC>",    //蓝牙 MAC 地址
  "BluetoothName": "<bluetooth name>",   //蓝牙名称
  "BluetoothAddCardCmd": "<data>",       //(将下发给蓝牙钥匙扣的)蓝牙指令数据, hex 串
  "UserID": <UserID>                  //锁用户 ID
}
}

```

【注】

同一把锁同 1 秒内只能获取 1 次(同 1 秒内多次调用返回结果相同);

若是纯蓝牙锁(或是 NB-IoT 网络不畅通的情形下), 为了让锁的系统时钟正确, 建议 App 与锁蓝牙通讯时调用: [蓝牙将服务器的系统时间同步到锁](#)

4.2.24. 蓝牙加卡用户

方法: post

请求头(header):

```

Accept-Language: <language>          //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.BluetoothAddCardUser",
  "deviceid": "<deviceid>",            //网关设备 ID
  "nodeid": <nodeid>,                 //智能锁在网关设备上的节点 id
  "params": {
    "UserPassword": "<UserPassword>", //卡数据:
                                          MF 卡, 4byte 值转换为十六进制串
                                          身份证卡, 8byte 值转换为十六进制串
    "UserID": <UserID>                 //锁用户 ID: 1~230 (未指定则由锁自动分配)
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1,          //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>",    //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>",   //蓝牙名称
    "BluetoothAddCardCmd": "<data>"        //蓝牙指令数据, hex 串
  }
}

```

【注】

同一把锁同 1 秒内只能获取 1 次(同 1 秒内多次调用返回结果相同);

若是纯蓝牙锁(或是 NB-IoT 网络不畅通的情形下), 为了让锁的系统时钟正确, 建议 App 与锁蓝牙通讯时调用: [蓝牙将服务器的系统时间同步到锁](#)

4.2.25. 蓝牙设置锁用户的有效星期时间

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothUpdateUserValidWeekTime",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
  "params": {
    "UserID": "<UserID>", //锁用户 ID
    "ValidWeekDays": "<SumWeekdayValue>", //有效星期累加值: 如 5 表示星期天和星期二有效
    // 1-星期天 2-星期一 4-星期二 8-星期三
    // 16-星期四 32-星期五 64-星期六
    "ValidWeekTime": "<HH:MM~HH:MM>" //有效星期的起止时间
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
  }
}
```

【注】

同一把锁同 1 秒内只能获取 1 次(同 1 秒内多次调用返回结果相同);

若是纯蓝牙锁(或是 NB-IoT 网络不畅通的情形下), 为了让锁的系统时钟正确, 建议 App 与锁蓝牙通讯时调用: [蓝牙将服务器的系统时间同步到锁](#)

4.2.26. 蓝牙设置锁用户的有效日期时间

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothUpdateUserValidDateTime",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
  "params": {
    "UserID": "<UserID>", //锁用户 ID
    "ValidBeginDateTime": "<YYYY-MM-DD HH:MM>", //有效期的开始日期时间
  }
}
```

```

        "ValidEndDateTime": "<YYYY-MM-DD HH:MM>" //有效期的终止日期时间
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
    "data": {
        "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
        "BluetoothName": "<bluetooth name>", //蓝牙名称
        "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
    }
}

```

【注】

同一把锁同 1 秒内只能获取 1 次(同 1 秒内多次调用返回结果相同);

若是纯蓝牙锁(或是 NB-IoT 网络不畅通的情形下), 为了让锁的系统时钟正确, 建议 App 与锁蓝牙通讯时调用: [蓝牙将服务器的系统时间同步到锁](#)

4.2.27. 蓝牙清除锁用户的有效日期时间

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.BluetoothClearUserValidDateTime",
    "deviceid": "<deviceid>", //网关设备 ID
    "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
    "params": {
        "UserID": <UserID> //锁用户 ID
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
    "data": {
        "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
        "BluetoothName": "<bluetooth name>", //蓝牙名称
        "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
    }
}

```

【注】

同一把锁同 1 秒内只能获取 1 次(同 1 秒内多次调用返回结果相同);

若是纯蓝牙锁(或是 NB-IoT 网络不畅通的情形下), 为了让锁的系统时钟正确, 建议 App 与锁蓝牙通讯时调用: [蓝牙将服务器的系统时间同步到锁](#)

4.2.28. 蓝牙删除锁用户

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothDeleteUser",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": "<nodeid>",               //智能锁在网关设备上的节点 id
  "params": {
    "UserID": "<UserID>"             //锁用户 ID
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,           //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
  }
}
```

【注】

同一把锁同 1 秒内只能获取 1 次(同 1 秒内多次调用返回结果相同);

若是纯蓝牙锁(或是 NB-IoT 网络不畅通的情形下), 为了让锁的系统时钟正确, 建议 App 与锁蓝牙通讯时调用: [蓝牙将服务器的系统时间同步到锁](#)

4.2.29. 获取待下发的锁用户异步请求

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.GetTodoLockUserAsyncInvoke",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": "<nodeid>"               //智能锁在网关设备上的节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,           //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
}
```

```

    "data": [
      {
        "_method": "<method>", //method:
                                thing.service.AddUser
                                thing.service.UpdateUserValidDateTime
                                thing.service.UpdateUserValidWeekTime
                                thing.service.DeleteUser
        "_params": {
          ...
        }
      },
      ...
    ]
  }
}

```

4.2.30. 蓝牙获取锁时间

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
 Authorization: Bearer <JWT token>

请求参数:

```

{
  "method": "thing.service.BluetoothGetTime",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>" //智能锁在网关设备上的节点 id
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
  }
}

```

4.2.31. 蓝牙获取锁用户列表

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
 Authorization: Bearer <JWT token>

请求参数:

```

{

```

```

"method": "thing.service.BluetoothGetUserList",
"deviceid": "<deviceid>", //网关设备 ID
"nodeid": <nodeid>, //智能锁在网关设备上的节点 id
"params": {
    "UserType": <UserType> //锁用户类型: 0-全部 1-密码 2-卡片 3-指纹
}
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
    "data": {
        "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
        "BluetoothName": "<bluetooth name>", //蓝牙名称
        "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
    }
}

```

4.2.32. 蓝牙获取锁用户类型

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.BluetoothGetUserType",
    "deviceid": "<deviceid>", //网关设备 ID
    "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
    "params": {
        "UserID": <UserID> //锁用户 ID(1~230)
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
    "data": {
        "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
        "BluetoothName": "<bluetooth name>", //蓝牙名称
        "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
    }
}

```

4.2.33. 蓝牙让锁进入 Z-Wave 添加/删除设备状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothEnterAddDelDeviceStatus",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid> //智能锁在网关设备上的节点 id
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
  }
}
```

4.2.34. 蓝牙清除锁数据

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothClearData",
  "async": false, //false-同步模式(缺省), true-异步模式
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
  "params": {
    "Mode": <Mode> // 0-恢复出厂 1-清空密码 2-清空卡片 3-清空指纹
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
  }
}
```

```

        "BluetoothAddCardCmd": "<data>" //蓝牙指令数据, hex 串
    }
}

```

4.3. 历史记录查询服务

url: http(s)://<ip>:<port>/app

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

4.3.1. (Redis5)查询设备 report

*设备事件 report 存储在 Redis5 的情况下才可使用此接口查询

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```

{
    "method": "thing.service.QueryDeviceReport",
    "deviceid": "<deviceid>", //网关设备 ID
    "nodeid": "<nodeid>", //节点 id
    "params": {
        "PageSize": "<pagesize>", //最大 100
        "MaxTimestamp": "<timestamp>", //Unix 时间戳(毫秒)
        "MinTimestamp": "<timestamp>", //Unix 时间戳(毫秒)
        "Type": "<event type>", //report 事件类型: notification、alarm、lockuser
        "Report": "<event name>", //report 事件名称
        "ExceptType": ["<event type>", "<event type>"],
        "ExceptReport": ["<event name>", "<event name>"]
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功, 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
    "data": {
        "Count": "<Count>", //返回记录数
        "NextTimestamp": "<timestamp>", //查询下一页时将此值传给 MaxTimestamp 参数
        "List": [
            {
                "id": "<id>",
                "ts": "<timestamp>", // Unix 时间戳(秒)
                "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
            }
        ]
    }
}

```

```

        "deviceid": "<deviceid>", //网关设备 ID
        "nodeid": "<nodeid>", //智能锁在网关设备上的节点 id
        "report": "<event name>", //report 事件名称
        "type": "<event type>", //report 事件类型
        "params": { //report 事件参数
            ...
        },
        "channelid": "<channelid>", //通道 id
        "sensortype": "<sensortype>", //传感器类型
        "logicnodeid": "<logicnodeid>", //逻辑节点 id
        "report_cn": "<report event name>", //本地化事件名称(Accept-Language 相关)
        "detail_cn": "<report event description>" //本地化事件描述(Accept-Language 相关)
    }
    ...
}
]
}

```

[示例]

curl -X GET \

http(s)://dev.isurpass.com.cn:8888/app \

-H 'Accept: */*' \

-H 'Authorization: Bearer

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \

-d '{

"method": "thing.service.QueryDeviceReport",

"deviceid": "iSurpass2018100000",

"nodeid": 2,

"params": {

"PageSize": 5,

"ExceptReport": ["thing.event.property.post"]

}

'

//返回结果:

```

{
  "resultcode": 1,
  "data": {
    "Count": 5,
    "NextTimestamp": 1565481230912,
    "List": [
      {
        "id": "1565482486619-0",
        "deviceid": "iSurpass2018100000",
        "nodeid": 2,
        "report": "thing.event.AddUser.post",
        "type": "lockuser",
        "params": {
          "UserID": 33,
          "UserLevel": 2,
          "UserLevel_CN": "normal",
          "UserName": "my fingerprint",
          "UserType": 2,
          "UserType_CN": "fingerprint"
        },
        "logicnodeid": "iSurpass2018100000-2-0-",
        "ts": 1566271471,

```

```

        "dt": "2019-08-20 11:24:31",
        "report_cn": "Add/Edit Lock User",
        "detail_cn": "Add/edit fingerprint user my fingerprint"
    },
    {
        "id": "1565482234612-0",
        "deviceid": "iSurpass2018100000",
        "nodeid": 2,
        "report": "thing.event.AddUser.post",
        "type": "lockuser",
        "params": {
            "UserID": 20,
            "UserLevel": 2,
            "UserLevel_CN": "normal",
            "UserName": "cardx",
            "UserPassword": "C7AF0BD6",
            "UserType": 3,
            "UserType_CN": "card"
        },
        "logicnodeid": "iSurpass2018100000-2-0-",
        "ts": 1566271219,
        "dt": "2019-08-20 11:20:19",
        "report_cn": "Add/Edit Lock User",
        "detail_cn": "Add/edit card user cardx"
    },
    {
        {
            "id": "1565482198977-0",
            "deviceid": "iSurpass2018100000",
            "nodeid": 2,
            "report": "thing.event.AddUser.post",
            "type": "lockuser",
            "params": {
                "UserID": 20,
                "UserLevel": 2,
                "UserLevel_CN": "normal",
                "UserPassword": "C7AF0BD6",
                "UserType": 3,
                "UserType_CN": "card"
            },
            "logicnodeid": "iSurpass2018100000-2-0-",
            "ts": 1566271183,
            "dt": "2019-08-20 11:19:43",
            "report_cn": "Add/Edit Lock User",
            "detail_cn": "Add/edit card user 20"
        },
        {
            "id": "1565481466642-0",
            "deviceid": "iSurpass2018100000",
            "nodeid": 2,
            "report": "thing.event.ClearNodeData.post",
            "type": "lockuser",
            "params": {
                "Mode": 0,
                "Mode_CN": "all"
            },
            "logicnodeid": "iSurpass2018100000-2-0-",
            "ts": 1566270451,

```

```

        "dt": "2019-08-20 11:07:31",
        "report_cn": "Clear Lock Users",
        "detail_cn": "Clear all users"
    },
    {
        "id": "1565481230913-0",
        "deviceid": "iSurpass2018100000",
        "nodeid": 2,
        "report": "thing.event.AddUser.post",
        "type": "lockuser",
        "params": {
            "UserID": 20,
            "UserLevel": 2,
            "UserLevel_CN": "normal",
            "UserName": "card a",
            "UserPassword": "C7AF0BD6",
            "UserType": 3,
            "UserType_CN": "card"
        },
        "logicnodeid": "iSurpass2018100000-2-0-",
        "ts": 1566270215,
        "dt": "2019-08-20 11:03:35",
        "report_cn": "Add/Edit Lock User",
        "detail_cn": "Add/edit card user card a"
    }
]
}

```

4.3.2. (InfluxDB)查询设备 report

*设备事件 report 存储在 InfluxDB V2 的情况下才可使用此接口

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.QueryDeviceReportFromTiDB",
    "deviceid": "<deviceid>",           //网关设备 ID, 可选参数
    "nodeid": "<nodeid>",               //节点 id, 可选参数
    "params": {
        "PageSize": "<pagesize>",      //缺省值: 10
        "Page": "<page>",              //缺省值: 1
        "Start": "<start>",             //缺省值: -1d(最近 1 天以内)
        "Stop": "<stop>",              //缺省值: now()(到目前为止)
        "Types": [<event type>],       //report 事件类型: notification、alarm、lockuser, 可选参数
        "Reports": [<event name>],     //report 事件名称, 可选参数
        "ParamsConditions": {"<param field>": "<params value>"} //report params, 可选参数
    }
}

```

```
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功, 非 1-失败
  "errorinfo": "",         //错误信息
  "errorinfo_cn": "",      //本地化错误信息(Accept-Language 相关)
  "data": {
    "Count": <Count>,      //返回记录数
    "List": [
      {
        "ts": <timestamp>, // Unix 时间戳(秒)
        "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
        "deviceid": "<deviceid>", //网关设备 ID
        "nodeid": <nodeid>, //智能锁在网关设备上的节点 id
        "report": "<event name>", //report 事件名称
        "type": "<event type>", //report 事件类型
        "params": {          //report 事件参数
          ...
        },
        "channelid": <channelid>, //通道 id
        "sensortype": "<sensortype>", //传感器类型
        "logicnodeid": "<logicnodeid>", //逻辑节点 id
        "report_cn": "<report event name>", //本地化事件名称(Accept-Language 相关)
        "detail_cn": "<report event description>" //本地化事件描述(Accept-Language 相关)
      }
    ]
  }
}
```

[示例]查询最近 3 天指定锁用密码/卡片/身份证开锁历史记录(成功/失败)

```
curl -X GET \
  http(s)://dev.isurpass.com.cn:8888/app \
  -H 'Accept: */*' \
  -H 'Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d {
    "method": "thing.service.QueryDeviceReportFromTiDB",
    "deviceid": "iRemote7005000000020",
    "nodeid": 38,
    "params": {
      "PageSize": 3,
      "Page": 1,
      "Start": "-3d",
      "Reports": ["thing.event.DoorOpenNotification.post", "thing.event.FailToOpenLock.post"],
      "ParamsConditions": {"Mode": [1,3,4]}
    }
  }
,
```

//返回结果:

```
{
  "resultcode": 1,
  "data": {
    "Count": 7,
    "List": [
```

```

    {
      "deviceid": "iRemote7005000000020",
      "nodeid": 38,
      "report": "thing.event.DoorOpenNotification.post",
      "type": "notification",
      "RawData":
"1E090031004700012600460016800700B11001030015060000000000000000130C1A1322006800045E049AE5004F0001
010056000100",
      "params": {
        "HappenedTime": "2019-12-26 19:34",
        "Happened_ts": 1577388840,
        "Mode": 1,
        "Mode_CN": "password",
        "UserID": 6,
        "UserName": "user6"
      },
      "logicnodeid": "iRemote7005000000020-38-0-",
      "ts": 1577360101,
      "dt": "2019-12-26 19:35:01",
      "report_cn": "unlocked",
      "detail_cn": "the user user6 opened the lock with password"
    },
    {
      "deviceid": "iRemote7005000000020",
      "nodeid": 38,
      "report": "thing.event.FailToOpenLock.post",
      "type": "notification",
      "RawData":
"1E090031004700012600460016800700B21001040015000000000000000000000130C1A131D006800045E0499B5004F000
1010056000100",
      "params": {
        "HappenedTime": "2019-12-26 19:29",
        "Happened_ts": 1577388540,
        "InvalidType": 0,
        "InvalidType_CN": "invalid user",
        "Mode": 1,
        "Mode_CN": "password"
      },
      "logicnodeid": "iRemote7005000000020-38-0-",
      "ts": 1577359797,
      "dt": "2019-12-26 19:29:57",
      "report_cn": "fail to open lock",
      "detail_cn": "invalid user"
    },
    {
      "deviceid": "iRemote7005000000020",
      "nodeid": 38,
      "report": "thing.event.DoorOpenNotification.post",
      "type": "notification",
      "RawData":
"1E090031004700012600460016800700B110010300150600000000000000000000130C1A131D006800045E0499AA004F000
1010056000100",
      "params": {
        "HappenedTime": "2019-12-26 19:29",
        "Happened_ts": 1577388540,
        "Mode": 1,
        "Mode_CN": "password",

```

```

        "UserID": 6
    },
    "logicnodeid": "iRemote7005000000020-38-0-",
    "ts": 1577359786,
    "dt": "2019-12-26 19:29:46",
    "report_cn": "unlocked",
    "detail_cn": "the user 6 opened the lock with password"
}
]
}
}
}

```

4.3.3. 查询异步请求

*异步请求记录依赖于 MongoDB 存储。

方法: post

请求头(header):

```

Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.QueryAsyncInvoke",
    "deviceid": "<deviceid>",           //网关设备 ID
    "nodeid": "<nodeid>",               //设备在网关上的节点 id
    "params": {
        "Conditions": {"<attr>":"<value>"}, //属性:值, 查询条件可以指定多个(与的关系)
        "SortBy": ["<attr>"],             //排序属性, attr 升序, -attr 降序, 排序可以指定多个
        "PageSize": "<pagesize>",         //每页记录数
        "Page": "<page>"                 //第几页
    }
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1,           //1-成功, 非 1-失败
    "errorinfo": "",          //错误信息
    "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
    "data": {
        "TotalCount": "<TotalCount>"    //总记录数
        "PageCount": "<PageCount>"      //总页数
        "PageSize": "<pagesize>",        //每页记录数
        "Page": "<page>",                //当前第几页
        "PageList": [              //当前页的设备列表
            {
                "_deviceid": "<deviceid>", //网关设备 ID
                "_nodeid": "<nodeid>",     //智能锁在网关设备上的节点 id
                "_method": "<method name>", //异步请求方法名
                "_params": {              //异步请求参数
                    ...
                }
            }
        ]
    }
}

```

```

    },
    "_processresult": "<resultcode>", //异步请求结果: 0-待下发, 1-成功, -9-取消, 其它值失败
    "_receipt": "<receipt>", //异步请求的回执 id
  }
  ...
]
}

```

4.3.4. (InfluxDB)查询远程请求日志 report

*远程请求日志存储在 InfluxDB V2 的情况下才可使用此接口

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.QueryRemoteInvokeLogFromTiDB",
  "deviceid": "<deviceid>", //网关设备 ID, 可选参数
  "nodeid": "<nodeid>", //节点 id, 可选参数
  "logicnodeid": "<logicnodeid>", //逻辑节点 id, 可选参数
  "params": {
    "PageSize": "<pagesize>", //缺省值: 10
    "Page": "<pagesize>", //缺省值: 1
    "Start": "<start>", //缺省值: -1d(最近 1 天以内)
    "Stop": "<stop>", //缺省值: now()(到目前为止)
    "By": "<by>", //远程请求方式: http/websocket
    "Method": "<method>" //远程请求方法
  }
}

```

4.4. TCP 网关节点设备管理服务

*TCP 网关的节点及其属性的存取依赖于 MongoDB

url: http(s)://<ip>:<port>/app/tcpgateway

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

4.4.1. 注册节点设备

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "deviceid": "<deviceid>", //TCP 网关 ID
  "method": "thing.service.RegistNode",
  "params": {
    "NodeID": "<nodeid>", //TCP 设备在 TCP 网关上的节点 id, 必须大于 1
    "Name": "<name>", //TCP 设备名称
    "IP": "<ip>", //TCP 设备 ip
    "Port": "<port>", //TCP 设备服务端口
    "DeviceModel": "<device model>", //TCP 设备型号
    "DeviceType": "<device type>", //TCP 设备类型
    "ProtocolType": "<protocol type>", //TCP 设备转换协议类型
    "Manufacturer": "<manufacturer>" //TCP 设备制造商
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 其它值-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/tcpgateway \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "deviceid": "iTCP2019000001",
    "method": "thing.service.RegistNode",
    "params": {
      "NodeID": 4,
      "Name": "xxx",
      "IP": "192.168.5.55",
      "Port": 8443,
      "DeviceModel": "authenmetric",
      "DeviceType": "facerecognition",
      "ProtocolType": "authenmetric",
      "Manufacturer": "authenmetric"
    }
  }
',
//返回结果:
{
  "resultcode": 1
}
```

4.4.2. 注销节点设备

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "deviceid": "<deviceid>", //TCP 网关 ID
  "method": "thing.service.UnRegistNode",
  "params": {
    "NodeID": "<nodeid>" //TCP 设备在 TCP 网关上的节点 id, 必须大于 1
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 其它值-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

[示例]

```
curl -X POST \
  http(s)://112.74.20.113:8888/app/tcpgateway \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcmVzcjkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLk88N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "deviceid": "iTCP2019000001",
    "method": "thing.service.UnRegistNode",
    "params": {
      "NodeID": 4
    }
  }
',
//返回结果:
{
  "resultcode": 1
}
```

4.5. 滚筒洗衣机高级服务

*洗衣机相关数据存储依赖于 MongoDB

url: http(s)://<ip>:<port>/app/washer

encoding: UTF-8

*生产环境使用 https

4.5.1. 蓝牙控制洗衣机

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothControl",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //洗衣机控制设备在网关上的节点 id, 固定值 1
  "params": {
    "Power": "<power>", //电源: 0-关闭 1-开启
    "Mode": "<mode>", //模式: 0-不设定 1-单脱水 5-快洗 8-标准洗 10-大物洗 12-加温洗
    //35-筒自洁 36-高温杀菌洗 37-羽绒服 38-洗鞋
    "DropWashingsTime": "<time>", //洗涤剂投放时间, 单位 100ms
    "DryTime": "<time>", //烘干时间: 0-不设定 16-15 分钟 32-30 分钟 48-60 分钟
    //64-90 分钟 80-120 分钟 96-150 分钟
    "DryMode": "<mode>", //烘干模式: 0-不设定 1-标准 2-特干 3-晾干 4-取消
    "Mode1Time": "<time>", //单脱水时间: 5~9 分钟
    "Mode5Time": "<time>", //快洗时间: 25~32 分钟
    "Mode8Time": "<time>", //标准洗时间: 35~42 分钟
    "Mode10Time": "<time>", //大物洗时间: 45~52 分钟
    "Mode12Time": "<time>" //加温洗时间: 55~60 分钟
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>", //蓝牙指令数据, hex 串
    "BluetoothCmdOK": "<data>" // hex 串, 用于 app 判断蓝牙指令是否成功
  }
}
```

[示例]

curl -X POST \

http(s)://dev.isurpass.com.cn:8888/app/washer \

-H 'Content-Type: application/json' \

-H 'Authorization:Bearer

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslInRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkBB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \

-d '{

"method": "thing.service.BluetoothControl",

"deviceid": "iSurpass2020020202",

"nodeid": 1,

"params": {


```
Q.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
-d '{
  "method": "thing.service.BluetoothReset",
  "deviceid": "iSurpass2020020202",
  "nodeid": 1
}'
//返回结果:
{
  "resultcode": 1,
  "data": {
    "BluetoothCmd": "F01149A83050D34B996A26939CC1EF4E94EE7304",
    "BluetoothCmdOK": "1949A83050D34B996A26939CC1EF10A3",
    "BluetoothMAC": "1E:98:6C:00:2D:64",
    "BluetoothName": "1E:98:6C:00:2D:64"
  }
}
```

4.5.3. 蓝牙获取洗衣机当前状态

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.BluetoothGetStatus",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": <nodeid>                 //洗衣机控制设备在网关上的节点 id, 固定值 1
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                   //1-成功 非 1-失败
  "errorinfo": "",                   //错误信息
  "errorinfo_cn": "",                //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>"           //蓝牙指令数据, hex 串
  }
}
```

[示例]

```
curl -X POST \
  http(s)://dev.isurpass.com.cn:8888/app/washer \
  -H 'Content-Type: application/json' \
  -H 'Authorization:Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClsluRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
  -d '{
    "method": "thing.service.BluetoothGetStatus",
    "deviceid": "iSurpass2020020202",
    "nodeid": 1
  }'
```

```
//返回结果:
{
  "resultcode": 1,
  "data": {
    "BluetoothCmd": "F01F49A83050D34B996A26939CC1EF4E94E14A3C",
    "BluetoothMAC": "1E:98:6C:00:2D:64",
    "BluetoothName": "1E:98:6C:00:2D:64"
  }
}
```

4.6. 波轮洗衣机高级服务

*洗衣机相关数据存储依赖于 MongoDB
 url: http(s)://<ip>:<port>/app/washer
 encoding: UTF-8
 *生产环境使用 https

4.6.1. 蓝牙控制洗衣机

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.BluetoothControl",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": "<nodeid>",               //洗衣机控制设备在网关上的节点 id, 固定值 1
  "params": {
    "Cmd": "<cmd>",                  //命令: 1-未定义 2-工作 3-预约 5-取消预约
    "Mode": "<mode>"                 //模式: 1-加强洗 2-标准洗 3-快速洗 4-单脱水 5-桶消毒 6-自检
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                    //1-成功 非 1-失败
  "errorinfo": "",                    //错误信息
  "errorinfo_cn": "",                 //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>",           //蓝牙指令数据, hex 串
    "BluetoothCmdOK": "<data>"         // hex 串, 用于 app 判断蓝牙指令是否成功
  }
}
```

4.6.2. 蓝牙获取洗衣机当前状态

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothGetStatus",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //洗衣机控制设备在网关上的节点 id, 固定值 1
  "params": {
    "Mode": "<mode>" //模式: 1-加强洗 2-标准洗 3-快速洗 4-单脱水 5-桶消毒 6-自检
  }
}
```

}

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>" //蓝牙指令数据, hex 串
  }
}
```

4.7. 波轮洗衣机 B 高级服务

*洗衣机相关数据存储依赖于 MongoDB

url: http(s)://<ip>:<port>/app/washer

encoding: UTF-8

*生产环境使用 https

4.7.1. 蓝牙控制洗衣机

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothControl",
  "deviceid": "<deviceid>",    //网关设备 ID
  "nodeid":    "<nodeid>",      //洗衣机控制设备在网关上的节点 id, 固定值 1
  "params": {
    "Cmd": "<cmd>",            //命令: 0: 无
                                1: 标准洗
                                2: 快速洗
                                3: 强力洗
                                4: 单脱水
                                5: [轻柔洗]
                                6: [漂洗并脱水]
                                7: 桶清洁1
                                8: [能效-全载]
                                9: [能效-半载]
                                10: [整机自检]
                                11: [电脑板自检]
    "StartKey": "<status val>",    //启动键状态: 0: 未启动 1: 启动洗衣 2: 停止洗衣
    "ExtraTime": "<minutes>",      //洗衣结束后的占用时间: 1~20(分钟)
    "WaterLevel": "<water level>"  //洗涤水位: 1~8 档
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,            //1-成功 非 1-失败
  "errorinfo": "",           //错误信息
  "errorinfo_cn": "",        //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>",    //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>",  //蓝牙名称
    "BluetoothCmd": "<data>",             //蓝牙指令数据, hex 串
    "BluetoothCmdOK": "<data>"           // hex 串, 用于 app 判断蓝牙指令是否成功
  }
}
```

4.7.2. 蓝牙获取洗衣机当前状态

方法: post

请求头(header):

```
Accept-Language: <language>    //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.BluetoothGetStatus",
  "deviceid": "<deviceid>",    //网关设备 ID
  "nodeid":    "<nodeid>"      //洗衣机控制设备在网关上的节点 id, 固定值 1
}
```

响应结果(Content-Type: application/json):

```
{
```

```

"resultcode": 1,          //1-成功 非 1-失败
"errorinfo": "",         //错误信息
"errorinfo_cn": "",      //本地化错误信息(Accept-Language 相关)
"data": {
  "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
  "BluetoothName": "<bluetooth name>", //蓝牙名称
  "BluetoothCmd": "<data>"           //蓝牙指令数据, hex 串
}
}

```

4.8. 洗鞋机高级服务

*洗鞋机相关数据存储依赖于 MongoDB

url: http(s)://<ip>:<port>/app/shoewasher

encoding: UTF-8

*生产环境使用 https

4.8.1. 蓝牙控制洗鞋机

方法: post

请求头(header):

```

Accept-Language: <language>          //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.BluetoothControl",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>,      //洗鞋机控制设备在网关上的节点 id
  "params": {
    "Mode1": <mode>, //模式: 1-大物洗 2-标准洗 4-快洗 8-单脱水 16-桶清洁 32-结束
                    64-暂停 128-恢复
    "Mode2": <mode>  //模式: 0-全关闭 1-自检模式 1 开启 2-自检模式 2 开启 4-跳转开启 8-结束开启
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>",           //蓝牙指令数据, hex 串
  }
}

```

```

    "BluetoothCmdOK": "<data>" // hex 串, 用于 app 判断蓝牙指令是否成功
  }
}

```

4.8.2. 蓝牙设定价格

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```

{
  "method": "thing.service.BluetoothSetPrice",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //洗鞋机控制设备在网关上的节点 id
  "params": {
    "Price3": <price>, //单脱价格
    "Price4": <price>, //快速价格
    "Price5": <price>, //标准价格
    "Price6": <price>, //大物价格
    "Price7": <price>, //加热快速价格
    "Price8": <price>, //加热标准价格
    "Price9": <price> //加热大物价格
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>", //蓝牙指令数据, hex 串
    "BluetoothCmdOK": "<data>" // hex 串, 用于 app 判断蓝牙指令是否成功
  }
}

```

4.8.3. 蓝牙设定时间参数

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```

{

```

```

"method": "thing.service.BluetoothSetTimeParameters",
"deviceid": "<deviceid>", //网关设备 ID
"nodeid": <nodeid>, //洗鞋机控制设备在网关上的节点 id
"params": {
  "Time3": <time>, //单脱时间
  "Time4": <time>, //快速时间
  "Time5": <time>, //标准时间
  "Time6": <time>, //大物时间
  "Time7": <time>, //液泵时间
  "Time8": <time>, //加热快速时间
  "Time9": <time>, //加热标准时间
  "Time10": <time> //加热大物时间
}
}
响应结果(Content-Type: application/json):
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>", //蓝牙指令数据, hex 串
    "BluetoothCmdOK": "<data>" // hex 串, 用于 app 判断蓝牙指令是否成功
  }
}

```

4.8.4. 蓝牙设定过程参数

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.BluetoothSetProgramParameters",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //洗鞋机控制设备在网关上的节点 id
  "params": {
    "Parameter3": <value>, //快速水位设定
    "Parameter4": <value>, //标准水位设定
    "Parameter5": <value>, //大物水位设定
    "Parameter6": <value>, //脱水转速设定
    "Parameter7": <value> //温度设定
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息

```

```

"errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
"data": {
  "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
  "BluetoothName": "<bluetooth name>", //蓝牙名称
  "BluetoothCmd": "<data>", //蓝牙指令数据, hex 串
  "BluetoothCmdOK": "<data>" // hex 串, 用于 app 判断蓝牙指令是否成功
}
}

```

4.8.5. 蓝牙设定特殊功能

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.BluetoothSetSpecialOptions",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid>, //洗鞋机控制设备在网关上的节点 id
  "params": {
    "SpecialOptions": <value>, //1-掉电记忆 2-全程保护 4-加热 8-桶清洁 16-按键 32-投币
  }
}

```

响应结果(Content-Type: application/json):

```

{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>", //蓝牙指令数据, hex 串
    "BluetoothCmdOK": "<data>" // hex 串, 用于 app 判断蓝牙指令是否成功
  }
}

```

4.8.6. 蓝牙发送键值

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
  "method": "thing.service.BluetoothSendKey",

```

```

    "deviceid": "<deviceid>", //网关设备 ID
    "nodeid": "<nodeid>", //洗鞋机控制设备在网关上的节点 id
    "params": {
        "Key": "<value>", //数值 0~9, 10-取消 11-确定
    }
}
}
响应结果(Content-Type: application/json):
{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
    "data": {
        "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
        "BluetoothName": "<bluetooth name>", //蓝牙名称
        "BluetoothCmd": "<data>", //蓝牙指令数据, hex 串
        "BluetoothCmdOK": "<data>" // hex 串, 用于 app 判断蓝牙指令是否成功
    }
}
}

```

4.8.7. 蓝牙获取洗鞋机当前状态

方法: post

请求头(header):

```

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

```

请求参数:

```

{
    "method": "thing.service.BluetoothGetStatus",
    "deviceid": "<deviceid>", //网关设备 ID
    "nodeid": "<nodeid>" //洗鞋机控制设备在网关上的节点 id, 固定值 1
}

```

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
    "data": {
        "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
        "BluetoothName": "<bluetooth name>", //蓝牙名称
        "BluetoothCmd": "<data>" //蓝牙指令数据, hex 串
    }
}
}

```

4.8.8. 蓝牙获取洗鞋机价格

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothGetPrice",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>" //洗鞋机控制设备在网关上的节点 id, 固定值 1
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>" //蓝牙指令数据, hex 串
  }
}
```

4.8.9. 蓝牙获取时间参数

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothGetTimeParameters",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>" //洗鞋机控制设备在网关上的节点 id, 固定值 1
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>" //蓝牙指令数据, hex 串
  }
}
```

4.8.10. 蓝牙获取过程参数

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.BluetoothGetProgramParameters",  
  "deviceid": "<deviceid>", //网关设备 ID  
  "nodeid": "<nodeid>" //洗鞋机控制设备在网关上的节点 id, 固定值 1  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "data": {  
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址  
    "BluetoothName": "<bluetooth name>", //蓝牙名称  
    "BluetoothCmd": "<data>" //蓝牙指令数据, hex 串  
  }  
}
```

4.8.11. 蓝牙获取特殊功能设定

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{  
  "method": "thing.service.BluetoothGetSpecialOptions",  
  "deviceid": "<deviceid>", //网关设备 ID  
  "nodeid": "<nodeid>" //洗鞋机控制设备在网关上的节点 id, 固定值 1  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
  "errorinfo": "", //错误信息  
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)  
  "data": {  
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址  
    "BluetoothName": "<bluetooth name>", //蓝牙名称  
    "BluetoothCmd": "<data>" //蓝牙指令数据, hex 串  
  }  
}
```

4.9. 干衣机高级服务

*洗衣机相关数据存储依赖于 MongoDB

url: http(s)://<ip>:<port>/app/dryer

encoding: UTF-8

*生产环境使用 https

4.9.1. 蓝牙控制干衣机

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothControl",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //干衣机控制设备在网关上的节点 id, 固定值 1
  "params": {
    "Cmd": "<hex>", //命令, 十六进制串
    "Select": "<hex>" //选择, 十六进制串
  }
}
```

//命令:

01H	启动运行	启动干衣机运行
02H	停止运行	停止干衣机工作
03H	开放按键	按键可以使用
04H	封锁按键	按键禁止使用
05H	温湿度查询	读取干衣机当前温湿度值
06H	桶自洁启动运行	桶自洁启动功能
07H	不启用智能判断	不启动智能模式, 按预置时间工作
08H	启用智能判断	自动判断干衣, 衣干即停

//干衣选择:

对应数值	发送数据含义
00H	干衣时间 30 分钟
01H	干衣时间 35 分钟
02H	干衣时间 40 分钟
...	...
4FH	干衣时间 425 分钟
50H	干衣时间 5 分钟
51H	干衣时间 10 分钟
52H	干衣时间 15 分钟
53H	干衣时间 20 分钟
54H	干衣时间 25 分钟
80H	干衣时间 30 分钟 + 杀菌
81H	干衣时间 35 分钟 + 杀菌
82H	干衣时间 40 分钟 + 杀菌
...	...
7FH	桶自洁收费模式
...	...
CFH	干衣时间 425 分钟 + 杀菌
D0H	干衣时间 5 分钟 + 杀菌
D1H	干衣时间 10 分钟 + 杀菌
D2H	干衣时间 15 分钟 + 杀菌
D3H	干衣时间 20 分钟 + 杀菌
D4H	干衣时间 25 分钟 + 杀菌

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,           //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>", //蓝牙指令数据, hex 串
    "BluetoothCmdOK": "<data>" // hex 串, 用于 app 判断蓝牙指令是否成功
  }
}
```

4.9.2. 蓝牙获取干衣机当前状态

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.BluetoothGetStatus",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": <nodeid> //干衣机控制设备在网关上的节点 id, 固定值 1
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>" //蓝牙指令数据, hex 串
  }
}
```

4.10. 其他高级服务

*扩展属性的存取依赖于 MongoDB

url: http(s)://**<ip>**:**<port>**/app

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

4.10.1. 强制删除设备

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN
Authorization: Bearer **<JWT token>**

请求参数:

```
{
  "method": "thing.service.ForceRemove",
```

```

    "deviceid": "<deviceid>" //网关设备 ID
}
响应结果(Content-Type: application/json):
{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}

```

【注】

调用该服务接口将强制清除指定设备的缓存以及 DB 存储数据，请谨慎使用！

[示例]

```

curl -X POST \
    http(s)://localhost/app \
    -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhcHBrcXkiOiJpU3VycGFzc0FwcClslRpbWVzdGFtcCI6IjEyMzQ1Njc4OTAifQ.KRLkB8N-3w-_tkUfUbHbZCBtZrHuOwKbmQeNK1w08_0' \
    -H 'Content-Type: application/json' \
    -H 'Postman-Token: 406f8f51-5bed-476e-ac82-dc0688df3b98' \
    -H 'cache-control: no-cache' \
    -d '{
        "method": "thing.service.ForceRemove",
        "deviceid": "iRemote2005000014683"
    }'
//返回结果:
{
    "resultcode": 1
}

```

4.11. 文件服务

4.11.1. 上传文件

url: http(s)://<ip>:<port>/app/uploadfile?token={JWT token}

encoding: UTF-8

*开发环境可以用 http，生产环境建议使用 https

方法: post

请求参数(multipart/form-data):

file

响应结果(Content-Type: application/json):

```

{
    "resultcode": 1, //1-成功 非 1-失败
    "errorinfo": "", //错误信息
    "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
}

```

```
    "FileURL": "<URL path>"           //文件的 URL Path
  }
}
```

【注】

上传的文件将存储在服务器的/opt/tools/iSurpass/upload/{appkey}目录下

4.12. 医药箱高级服务

*洗衣机相关数据存储依赖于 MongoDB

url: http(s)://<ip>:<port>/app/medicalbox

encoding: UTF-8

*生产环境使用 https

4.12.1. 获取蓝牙开锁密码

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
  "method": "thing.service.GetBluetoothKey",
  "deviceid": "<deviceid>",           //网关设备 ID
  "nodeid": "<nodeid>"                //节点 id, 固定值 1
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,                    //1-成功 非 1-失败
  "errorinfo": "",                    //错误信息
  "errorinfo_cn": "",                 //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>"           //蓝牙指令数据, hex 串
  }
}
```

4.12.2. 蓝牙控制设备指令

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.BluetoothControlCmd",
  "deviceid": "<deviceid>", //网关设备 ID
  "nodeid": "<nodeid>", //节点 id, 固定值 1
  "params": {
    "Mode": "<mode>", //模式:
    // 0: 关
    // 1: 开
    "PointTemperature": "<value>" //设定温度(2~18)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
  "data": {
    "BluetoothMAC": "<bluetooth MAC>", //蓝牙 MAC 地址
    "BluetoothName": "<bluetooth name>", //蓝牙名称
    "BluetoothCmd": "<data>" //蓝牙指令数据, hex 串
  }
}
```

4.13. 批处理

url: http(s)://<ip>:<port>/app/batch

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

4.13.1. 执行批处理

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>

请求参数:

```
{
  "method": "thing.service.dobatch",
  "params": {
    "BatchID": "<id>", //批处理任务 id(字母或数字, 最长 10 byte)
    "Mode": "<serial/parallel>", //serial:串行, parallel:并行
    "List": [
      {
        "Command": "<json>", //第 2 章节的设备远程控制(部分)、或是嵌套一个批处理
        "Delay": "<seconds>" //延时时间(0~30 秒)
        // Mode 值为 parallel 时, 延时相对于批处理开始时间
        // Mode 值为 serial 时, 延时相对于上一个 command 的结束时间
      },
      ...
    ]
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-已开始执行 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

【注】

- 1) resultcode 返回 1 仅表示批处理已开始执行, 不表示批处理已完成/成功/失败
- 2) 批处理的最大处理时长是 600 秒, 超过会自动中止
- 3) 如若需要接收批处理推送:
 - a) WebSocket 方式, 参见 3.9
 - b) BeeHive HTTP 推送方式
- 4) 支持的设备远程控制命令如下:
 - 网关情景: [执行网关情景](#)
 - 锁: [远程开锁](#)、[远程落锁](#)
 - 开关: [远程开](#)、[远程关](#)
 - 窗帘: [远程开](#)、[远程开\(指定百分比\)](#)、[远程关](#)
 - 调光灯: [远程开](#)、[远程开\(指定百分比\)](#)、[远程关](#)
 - 空调: [远程开](#)、[远程关](#)、[远程设置模式](#)、[远程设置温度](#)、[远程设置风速](#)
 - 红外控制: [远程开](#)、[远程关](#)、[远程控制](#)
 - 多媒体主机: [远程打开视频](#)、[远程打开 PPT](#)、[远程热键控制](#)
 - 串口设备: [远程开](#)、[远程关](#)、[运行情景](#)、[远程控制](#)

4.13.2. 中止批处理

方法: post

请求头(header):

```
Accept-Language: <language> //en、zh-CN
Authorization: Bearer <JWT token>
```

请求参数:

```
{
```

```
    "method": "thing.service.cancelbatch",
    "params": {
        "BatchID": "<id>" //批处理任务 id
    }
}
```

响应结果(Content-Type: application/json):

```
{
    "resultcode": 1, //1-已开始执行 0-指定批处理不存在或已结束
    "errorinfo": "", //错误信息
    "errorinfo_cn": "" //本地化错误信息(Accept-Language 相关)
}
```

5. (4G CAT1)人脸设备

*依赖于 4G CAT1 人脸设备专用服务程序以及 MongoDB 存储

5.1. http 基础服务

http(s)://<ip>:<port>/face4g

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

5.1.1. 添加人员

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

请求参数:

```
{
  "method": "thing.service.AddUser",
  "deviceid": "<mac>", //人脸设备 MAC
  "params": {
    "name": "<name>", //人员姓名
    "image": "<base64 data>" //人员相片 base64 数据
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "errorinfo": "", //错误信息
  "errorinfo_cn": "", //本地化错误信息(Accept-Language 相关)
```

```
"result": {
  "uid": "<uid>"           //人脸 uid(32 byte 长)
}
```

5.1.2. 删除人员

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

请求参数:

```
{
  "method": "thing.service.DeleteUser",
  "deviceid": "<mac>",           //人脸设备 MAC
  "params": {
    "uid": "<uid>"               //人脸 uid(32 byte 长)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,           //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
}
```

5.1.3. 清空人员

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

请求参数:

```
{
  "method": "thing.service.ClearUser",
}
```

```
"deviceid": "<mac>"           //人脸设备 MAC
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,           //1-成功 非 1-失败
  "errorinfo": "",          //错误信息
  "errorinfo_cn": "",       //本地化错误信息(Accept-Language 相关)
}
```

5.1.4. 获取人员列表

方法: post

请求头(header):

```
Accept-Language: <language>           //en、zh-CN
```

请求参数:

```
{
  "method": "thing.service.GetUserList",
  "deviceid": "<mac>"           //人脸设备 MAC
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,           //1-成功 非 1-失败
  "result": [
    {
      "uid": "<uid>",          //人脸 uid(32 byte 长)
      "name": "<name>"         //人员姓名
    }
    ...
  ]
}
```

5.1.5. 获取人员名称

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN

请求参数:

```
{
  "method": "thing.service.GetUserName",
  "deviceid": "<mac>", //人脸设备 MAC
  "params": {
    "uid": "<uid>" //人脸 uid(32 byte 长)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "result": {
    "name": "<name>" //人员姓名
  }
}
```

5.1.6. 获取人员相片

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN

请求参数:

```
{
  "method": "thing.service.GetUserImage",
  "deviceid": "<mac>", //人脸设备 MAC
  "params": {
    "uid": "<uid>" //人脸 uid(32 byte 长)
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
  "result": {
    "image": "<base64 data>" //人员相片 base64 数据
  }
}
```

```
}  
}
```

5.1.7 下发卡

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN

请求参数:

```
{  
  "method": "thing.service.SetCard",  
  "deviceid": "<mac>", //人脸设备 MAC  
  "params": {  
    "type": "<type>" //卡类型为 nfc、wgd、433  
    "cardnumber": "<cardnumber>"  
  }  
}  
// nfc 卡的值, 数值为 16 进制的字符串, 长度为 4 字节 (A 卡) 或者 10 字节 (B 卡)  
eg: "cardnumber": ["2a818516", "11223344556677889900"]  
// 韦根卡号值数组, 如卡号为: 0004662575 071,09519, 只需要把 0004662575 传入即可 val 为数值类型,  
不需要把前面的 0 传入  
eg: "cardnumber": [4662575, 13363568]  
// 433 卡号一般是 4 字节, 要是自定义数据是无法确定长度, 数值是 16 进制的字符串  
eg: "cardnumber": ["11223344", "aabbccdd"]  
}  
}
```

响应结果(Content-Type: application/json):

```
{  
  "resultcode": 1, //1-成功 非 1-失败  
}
```

5.1.8 绑定蓝牙门锁

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN

请求参数:

```
{
  "method": "thing.service.SetBleLock",
  "deviceid": "<mac>", //人脸设备 MAC
  "params": {
    "lock_mac": [{"mac": "<mac>", "data": "<data>"}...]
  }
}
```

eg: "cardnumber": [{"mac": "112233445566", "data": "aabbcc"}, {"mac": "aabbccddeeff", "data": "aabbcc"}]

mac 为蓝牙门锁的 mac 地址, 6 字节, 即 12 字符, 数值是 16 进制的字符串。(去掉冒号)

data 为蓝牙的透传数据(或控制数据), 可选字段, 数值是 16 进制的字符串, 若实现刷脸开锁, data 为"0101"。

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
}
```

5.1.9 蓝牙透传数据

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

请求参数:

```
{
  "method": "thing.service.BleCardValue",
  "deviceid": "<mac>", //人脸设备 MAC
  "params": {
    "mac": "112233445566", //门锁蓝牙 mac 地址。(去掉冒号)
    "value": "<value>" //透传的内容, 16 进制字符串, 设备发送到蓝牙模块时, 需要转成对应的 16 进制
    开锁为"0101"。
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
}
```

5.1.10 下发取电开关

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

请求参数:

```
{
  "method": "thing.service.SetPower",
  "deviceid": "<mac>", //人脸设备 MAC
  "params": {
    "card": "<card>", //card 为卡号
    "value": <value>, //value 为取电值 (0: 关闭取电, 1: 打开取电)
    "start_time": "YYYY-MM-DD HH:MM ", //起始时间
    "end_time": "YYYY-MM-DD HH:MM " //结束时间
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1, //1-成功 非 1-失败
}
```

5.1.11 冻结/解冻用户

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

请求参数:

```
{
  "method": "thing.service.FreezeUser",
  "deviceid": "<mac>", //人脸设备 MAC
  "params": {
    "value": <value> // `value 为 1, 表示解冻用户`value 为 0, 表示冻结用户`
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
}
```

5.1.12 更改休眠壁纸

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

请求参数:

```
{
  "method": " thing.service.UpdateResource",
  "deviceid": "<mac>",          //人脸设备 MAC
  "params": {
    "image": "<image>" // `为图片的 base64 数据，图片小于 16K 且分辨率为 240X320`
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
}
```

5.1.13 修改人脸配置

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

请求参数:

```
{
  "method": " thing.service.SetConfig",
  "deviceid": "<mac>",          //人脸设备 MAC
  "params": {
    "config": "<config>" //所有配置见下方
    "value": value //配置值
  }
}
```

```
}
- face_threshold: 识别阈值, 100 倍换算(0-100)X100(一百倍换算)
- led_bright: 白光灯 2 挡调光值, 100 倍换算(0-1)X100(一百倍换算)
- led_bright_low: 白光灯 1 挡调光值, 100 倍换算(0-1)X100(一百倍换算)
- living_threshold: 活体阈值, 100 倍换算 (0-100)X100(一百倍换算)
- relay_open_vol: 继电器打开电平(0 开启, 1 关闭)
- relay_open_s: 继电器打开时间(1s-3s)
- chk_face_pose: 检查人脸是否正视(0 开启, 1 关闭)
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功 非 1-失败
}
```

5.2. http 高级服务

http(s)://<ip>:<port>/app/face4g

encoding: UTF-8

*开发环境可以用 http, 生产环境建议使用 https

5.2.1. 设置 owner 和 name

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

请求参数:

```
{
  "method": "thing.service.SetOwner",
  "deviceid": "<mac>",           //人脸设备 MAC
  "params": {
    "owner": "<owner>",           //人脸设备的 owner
    "name": "<name>"              //人脸设备名称
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功, -1-设备已有 owner
  "errorinfo": "",      //错误信息
  "errorinfo_cn": ""    //本地化错误信息, 与 Accept-Language 相关
}
```

5.2.2. 设置 name

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN

请求参数:

```
{
  "method": "thing.service.SetName",
  "deviceid": "<mac>",      //人脸设备 MAC
  "params": {
    "name": "<name>"        //人脸设备名称
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,      //1-成功, 非 1-失败
  "errorinfo": "",      //错误信息
  "errorinfo_cn": ""    //本地化错误信息, 与 Accept-Language 相关
}
```

5.2.3. 获取人脸设备列表

方法: post

请求头(header):

Accept-Language: **<language>** //en、zh-CN

请求参数:

```
{
  "method": "thing.service.GetByOwner",
  "params": {
    "owner": "<owner>",          //人脸设备的 owner
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功 非 1-失败
  "result": [
    {
      "deviceid": "<mac>",          //人脸设备 MAC
      "name": "<name>",            //人脸设备名称
      "onlinestatus": <status>    //人脸设备上线状态: 0-未入网 1-上线 -1-离线
    }
    ...
  ]
}
```

5.2.4. 清除 owner

方法: post

请求头(header):

Accept-Language: <language> //en、zh-CN

请求参数:

```
{
  "method": "thing.service.ClearOwner",
  "deviceid": "<mac>",          //人脸设备 MAC
  "params": {
    "owner": "<old owner>"      //人脸设备的 owner
  }
}
```

响应结果(Content-Type: application/json):

```
{
  "resultcode": 1,          //1-成功, -1-owner 不正确
  "errorinfo": "",         //错误信息
}
```

```
    "errorinfo_cn": ""      //本地化错误信息，与 Accept-Language 相关
}
```

5.3. 人脸设备 report

5.3.1. 上线 report

```
{
  "deviceid": "<mac>",          //人脸设备 MAC
  "report": "thing.event.Online",
  "type": "notification",
  "params": {
    "DeviceType": "face",      //设备类型
    "ProtocolType": "face_cat1" //转换协议类型
  },
  "ts": <timestamp>,          //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称，与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述，与 Accept-Language 相关
}
```

5.3.2. 离线 report

```
{
  "deviceid": "<mac>",          //人脸设备 MAC
  "report": "thing.event.Offline",
  "type": "notification",
  "ts": <timestamp>,          //Unix 时间戳(秒)
  "dt": "<yyyy-mm-dd HH:MM:SS>", //日期时间
  "report_cn": "<report event name>", //本地化事件名称，与 Accept-Language 相关
  "detail_cn": "<report event description>" //本地化事件描述，与 Accept-Language 相关
}
```

5.3.3. 通行 report

```
{
  "deviceid": "<mac>",          //人脸设备 MAC
```

```

"report": "thing.event.FaceRecognition.post",
"type": "notification",           //notification-通知
"params": {
  "uid": "<uid>"                   //通行人员的 uid
  "veiryTime": "yyyy-mm-dd HH:MM:SS" //通行日期时间
},
"ts": <timestamp>,               //Unix 时间戳(秒)
"dt": "<yyyy-mm-dd HH:MM:SS>",    //日期时间
"report_cn": "<report event name>", //本地化事件名称, 与 Accept-Language 相关
"detail_cn": "<report event description>" //本地化事件描述, 与 Accept-Language 相关
}

```

5.4. 人脸设备 mongodb 存储

5.4.1. 人脸设备 facedevice

deviceid	//设备 id: MAC 地址
protocoltype	//协议类型: face_cat1
latest_sha256	//最新的人脸 sha256 值
owner	//设备 owner
name	//设备名称
onlinestatus	//上线状态: 0-未入网 1-上线 -1-离线

5.4.2. 人脸特征 facefeatures

deviceid	//设备 id: MAC 地址
uid	//人脸 uid
feature	//人脸特征码

A. 设备类型及其转换协议一览表

类型	nodetype	子类型	protocoltype	备注
锁	lock	Z-Wave 锁	lock	
		NB-IoT 锁	nblock	NB-IoT 设备
			nblock_2k	NB-IoT 设备 支持 2k 指纹
		WIFI 锁	wifilock	
			wifilock2	新版 wifi 锁
指纹采集器	fingerprintreader	iSurpass 指纹采集器	fingerprintreader	
开关	switch	Z-Wave 开关/插座/机械手臂/报警器	switch	
无源开关	switch	iSurpass 无源开关	natural	需结合支持无源开关的 Z-Wave 网关使用
情景面板	scenepanel	Z-Wave 情景面板	scenepanel	
窗帘	curtain	Z-Wave 窗帘电机	multilevelswitch	
调光灯	dimmer	Z-Wave 调光灯	multilevelswitch	
门磁窗磁	doorwindowsensor	Z-Wave 门窗磁	doorwindowsensor	
		NB-IoT 门窗磁	nbdoorwindowsensor	NB-IoT 设备
		NB-IoT 门窗磁 V2	nbdoorsensor2	
移动感应	motion	Z-Wave 移动感应	motion	
空调	aircondition	Z-Wave 空调	aircondition	
温度计	temperaturesensor	Z-Wave 温度计	temperaturesensor	
亮度计	luminancesensor	Z-Wave 亮度计	luminancesensor	
湿度计	humiditysensor	Z-Wave 湿度计	humiditysensor	
烟感	smokesensor	Z-Wave 烟感	smokesensor	
漏水感应	watersensor	Z-Wave 漏水感应	watersensor	
红外控制	irremote	Z-Wave 红外控制	irremote	
电表	electricmeter	Z-Wave 电表	electricmeter	
		OneNet 电表	electricmeter	NB-IoT 设备
水表	watermeter	Z-Wave 水表	watermeter	
		OneNet 水表	watermeter	NB-IoT 设备
人脸识别	facerecognition			
音视频	av	音丽士	levoice	需结合 TCP 网关使用
中控主机	pcmedia	iSurpass 中控主机	pcmedia	需结合 TCP

				网关使用
净水器	waterfilter	艾欧净水器	nbwaterfilter	NB-IoT 设备
洗衣机	washer	创维滚筒洗衣机	nbwasher	NB-IoT 设备
		创维波轮洗衣机	nbwasher_impeller	NB-IoT 设备
		创维波轮洗衣机 B	nbwasher_impeller_b	NB-IoT 设备
洗鞋机	shoewasher	创维洗鞋机	nbshoewasher	NB-IoT 设备
干衣机	dryer	创维干衣机	nbdryer	NB-IoT 设备
SOS	sos	iSurpass SOS	nbsos	NB-IoT 设备
		Z-Wave SOS	sos	
串口设备	serial	Z-Wave 串口设备	serial	
耳标	eartag		bteartag	蓝牙设备
燃气感应	gassensor	Z-Wave 燃气感应	gassensor	
路灯	roadlamp	路灯	nbroadlamp	NB-IoT 设备
医药箱	medicalbox		medicalbox	4G 设备
门禁	lock	门禁	entrance	需结合 TCP 网关使用
	lock	二维码	entranceb	
	lock	人脸	uniwin	
电梯	lift	电梯	lift	
sip 设备	sip	电梯对讲设备	sip	
断路器	breaker	智能断路器	breaker	需结合 TCP 网关使用

【注】

新版门磁(V2)需要 openservice-telecom v3.2.5 以上版本、openservice v2.2.0 以上版本(或 js 目录下有 nbdoorsensor2.js 文件)