镇江市物联网设备接入中间件

开发规范

本规范适用于第三方厂家的设备通过 MQTT 中间件接入管理平台。接入用中间件由厂家按照本协议规范进行开发。

1、名词定义

下行：指管理平台向厂家中间件主动发起请求，包括但不限于：参数设置，参数读取，设备控制，状态读取等。

推送：指厂家中间件向管理平台推送设备的应答数据，或者主动上报设备数据。

2、交互方式

上下行均采用 MQTT 方式进行交互，数据本体采用 json 格式，可依据需要对 json 字符串进行额外的编码。

设备无法返回的数据应使用零值或空字符串，不可使用 null。

所有字符编码均采用 utf-8 格式。

3、交互协议

3.1、MQTT Topic 格式定义

<数据方向>/<数据格式>/<厂家 ID>/<指令>/<设备标识，上行可选>

3.1.1、Topic 字段说明

- 数据方向：
  - down：下行，用于管理平台向厂家平台发送指令
  - up：上行，用于厂家平台向管理平台推送设备应答或主报数据
- 数据格式：基础格式为 json，其他格式均为对 json 字符串进行二次编码
  - json：body 内的数据采用 json 编码
  - base64：body 内数据先采用 json 编码，然后再使用 base64 编码的字符串
  - rsa：body 内数据先采用 json 编码，然后采用 rsa 算法加密的 base64 字符串（rsa 密钥由管理平台提供）
  - sm2：body 内数据先采用 json 编码，然后采用 sm2 算法加密的 base64 字符串（sm2 密钥由管理平台提供）
- 厂家 ID：由管理平台定义

- 命令 ID：用于指定读取/设置/控制的具体内容，如读取数据，读取时钟等等，具体参考 交互类指令协议
- 设备标识：用于标识设备的唯一 id，具体定义见 3.2 中的 context.dev_id 说明，上行时可不提供。

3.1.2、Topic 示例

- 例 1：读取厂家 1（厂家 id 为 01）终端设备的时钟
  - 下行：down/json/01/getTimer/010000002934857620
  - 上行： up/json/01/getTimer/010000002934857620

3.2、Body 格式定义

字段名	类型	是否必填	说明
context	json object	Y	指令标识内容，应答推送时应和下行保持一致
context.dev_attr	int	Y	设备支持的功能模组，定义见下表， **上行必填**，按照设备实际支持的模组功能设置
context.dev_id	string	Y	设备标识，用于标识设备的唯一 id 定义规则：总长固定为 18 位 ascII 码字符串，从左往右 头两位为识别 ID（申请接入时分配）， 后 16 位为设备 ID，厂家定义，不足的补'0'。 例： 厂家 A 设备 id： 010000002934857620

字段名	类型	是否必填	说明
			厂家 B 设备 id： 020000000000000001
context.gateway_id	string	N	设备连接的网关 id，用于 485 转发等传输场合
context.cmd	string	Y	指令
context.unix_milli	int64	Y	数据发生的时间戳，unix 格式， **精确到毫秒**
context.serial_id	string	N	用于标识请求的 id。 **当下行指令数据中包含该字段时，** **厂家中间件在进行应答数据的推送时也应包含该字段，** **且内容必须与下行一致。** 主动上报数据的推送可不提交该字段。
context.status	int	N	仅上行应答有， 0-设备响应失败， 1-设备响应成功， 2-设备不在线， 3-设备不支持该协议
context.detail	string	N	仅上行应答有， 当 status==0 时，填充失败原因
data	json object	Y	数据结构，具体定义见每个指令章节

- 设备模组定义说明

  设备模组为二进制字节，转换为 int，按位定义功能，具体定义如下：

bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0

位定义标识	模组功能
bit0	具备回路控制功能以及进出线数据采集功能
bit1	具备漏电检测/分闸功能
bit2	具备能耗数据采集功能
bit3	具备照度数据采集功能
bit4	回路控制具备调光功能（需配合 bit0 设置）
bit5	回路具备供电管理功能（需配合 bit0 设置）
bit6	具备电磁阀控制功能
bit7	保留

3.2.1、下行 Body 示例

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    ...
  }
}
```

3.2.2、上行 Body 示例

```
{
  "context":{
```

```
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":0,
    "detail":"device not answer"
  },
  "data":{
    ...
  }
}
```

3.3、设置类指令协议

3.3.1、设置设备通信参数

- 命令 ID：setCommParam
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "ip":"123.123.123.123", // 服务端 IP，string
    "port":1234, // 服务端端口，int：1024～65535
```

```
    "apn":"cmnet", // apn，string 4G 网络用
    "username":"abc", // 用户名，string
    "password":"def", // 密码，string
  }
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{}
}
```

3.3.2、设置时钟

- 命令 ID：setTimer
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
```

```
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "now":123456789 // 当前时间，unix 时间戳，精确到秒，int64
  }
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{}
}
```

3.3.3、设置开关灯时间

当时间设置较为细致，如一年内每天都不同，此时因为数据量过大，可能将内容拆分后下发。

二级平台收到指定日期的开关灯时间后，可先清除**当天**所有回路的控制时间，然后依据收到的时间，写入设备。

例 1：如果某天的 do 为空，则清空当天的所有开关灯设置，即当天不开灯

例 2：如果某天的 do 内只有 switch_id:3 的设置，则表示当天只对输出 3 进行操作，其他输出如果有设置值，应清除。

- 命令 ID：setTimetable
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID
  },
  "data":{
    "timetable":[{
      "dt":"07-01", // 执行日期，mm-dd 格式，string, 不填，表示每日均执行相同指令
      "do":[{
        "switch_id":1, // 接触器序号，int，从 1 开始
        "type":1, // 指令类型，int：1-开灯，2-关灯，3，调光，4-经纬度开，5-经纬度关
        "time":"19:00", // 指令执行时间，24 小时制，string（type=1/2/3）
        "dimming":80 // 调光比例，仅当`type==3`时有效，int：0～100（type=1/2/3）
        "offset":30 // 开关灯偏移量，int（-127～128）,分钟（type=4/5 时有效）
      },...]
```

```
        },...]
    }
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{}
}
```

3.3.4、设置漏电报警参数

设备因漏电而导致分匝后，需要对指定回路发送漏电消除指令后，才能执行开关灯指令。

- 命令 ID：setLeakageCurrent
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
```

```
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "leak":[{
      "switch_id":1, // 该回路对应的接触器序号，int，从 1 开始
      "baseline":20, // 基线（屏蔽值），毫安，int，小于等于基线值的数据忽略
      "alarm_limit_1":300, // 一级漏电报警值，毫安，int：设置为 0 时表示不使用漏电报警功能
      "alarm_limit_2":300, // 二级漏电报警值，毫安，int：设置为 0 时表示不使用漏电报警功能
      "alarm_limit_3":300, // 三级漏电报警值，毫安，int：设置为 0 时表示不使用漏电报警功能
      "break_limit":500, // 漏电分匝值，毫安，int：设置为 0 时表示不使用漏电分匝功能
      "ratio": 100, // 互感比，int, 100/5 填 100,其他依次类推
      "work_model":1 // 工作模式，int,0-不启用，1-只检测，2-检测并报警，3-检测不报警但分闸，4-检测既报警又分闸
    },...]
  }
}
```

- 推送数据：

```
{
  "context":{
  "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
```

```
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{}
}
```

3.3.5、设置回路参数

- 命令 ID：setLoopParam
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "energy":{ // 电能参数，无电能功能的设备可以不填
      "ratio_a":50, // a 相电能互感比，50:5 填 50，int
      "ratio_b":50, // b 相电能互感比，50:5 填 50，int
      "ratio_c":50, // c 相电能互感比，50:5 填 50，int
    },
    "voltage":{ // 电压限值参数
      "up_limit":250.00, // 过压报警值，float
      "up_limit_dispel":220.00, // 过压报警消除值，float
```

"low_limit":200.00, // 欠压报警值，float

"low_limit_dispel":220.00 // 欠压报警消除值，float

},

"loop":[{ // 回路参数以及限值

"loop_id":1, // 电流检测回路序号，从 1 开始，int

"switch_id":1, // 该回路对应的输出接触器序号，从 1 开始，0-表示不对应输出接触器，int

"phase":1, // 该回路对应的相位，1-A 相，2-B 相，3-C 相，int

"hopping":1, // 跳变报警（0-不报警 1-报警），int

"ratio",50, // 回路变比，50:5 填 50，int

"levels":[{ // 多个时间段设置不同时，提交多段

"dt_start":"20:00", // 报警起始时间，24 小时制，string，全天有效时，设置为`00:00`

"dt_end":"22:00", // 报警结束时间，24 小时制，string，全天有效时，设置为`24:00`

"up_limit":23.4, // 过流报警值，float

"up_limit_dispel":20.00, // 过流报警消除值，float

"low_limit":4.1, // 欠流报警值，float

"low_limit_dispel":8.00 // 欠流报警消除值，float

}...],

},...]

}

}

- 推送数据：

{

"context":{

"dev_attr":3, // 设备模组,

```
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{}
}
```

3.3.6、设置运行参数

- 命令 ID：setRunningParam
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID
  },
  "data":{
    "time_ticker":{
      "heartbeat":60, // 心跳周期，单位秒，int，0-不主动上报，默认 60
      "data":1800, // 数据主动上报周期，单位秒，int，0-不主动上报，默认 1800
      "lux":15 // 光照度数据推送周期，单位秒，int,0-不主动上报，
```

默认 15

```
      },
    "coord":{ // 经纬度数据，wgs84 数据
      "lng":123.32, // 经度，float
      "lat":32.32 // 纬度，float
      },
    "others":{
      "max_switch":1 // 最大控制回路数量
    }
  }
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{}
}
```

3.3.7、设置调光参数

- 命令 ID：setDimmingParam
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID
  },
  "data":{
    "dimming":[{
      "switch_id":1, // 回路接触器序号，int，从 1 开始
      "power_on_default":0, // 默认上电动作状态，int, 1-上电开灯，
0-上电关灯
      "rated_power":250, // 灯具额定功率，int，单位 W，0-不设置
      "dimming_type":0, // 调光方式，int，0-正向调光，1-反向调光
    },...]
  }
}
```

- 推送数据

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
```

```
    "detail":""
  },
  "data":{}
}
```

3.4、读取类指令

3.4.1、读取设备通信参数

- 命令 ID：getCommParam
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
```

```
    "status":1,
    "detail":""
  },
  "data":{
    "ip":"123.123.123.123", // 服务端 IP/域名，string
    "port":1234, // 服务端端口，int：1024～65535
    "apn":"cmnet", // apn，string
    "username":"abc", // 虚拟网用户名，string
    "password":"def", // 虚拟网密码，string
  }
}
```

3.4.2、读取设备时钟

- 命令 ID：getTimer
- 下行数据格式：

```
{
  "context":{
  "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据：

```
{
  "context":{
```

```
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{
    "now":123456789 // 当前时间，unix 时间戳，精确到秒，int64
  }
}
```

3.4.3、读取开关灯时间

当时间设置较为细致，如一年内每天都不同，此时因为数据量过大，可以将内容拆分后上送。

mqtt 的一包数据量建议控制在 1M 左右

- 命令 ID：getTimetable
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
```

```
    "begin_date":"07-01",// 要查询的开始日期，mm-dd 格式，string
    "end_date":"08-01" // 要查询的结束日期，mm-dd 格式，string
  }
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{
    "timetable":[{
      "dt":"07-01", // 执行日期，mm-dd 格式，string
      "do":[{
        "switch_id":1, // 接触器序号，int，从 1 开始
        "type":1, // 指令类型，int：1-开灯，2-关灯，3，调光，4-经纬度开，5-经纬度关
        "time":"19:00", // 指令执行时间，24 小时制，string（type=1/2/3）
        "dimming":80 // 调光比例，仅当`type==3`时有效，int：0～100（type=1/2/3）
        "offset":30 // 开关灯偏移量，int（-127～128）,分钟
```

（type=4/5 时有效）

```
        },...]
    },...]
  }
}
```

3.4.4、读取漏电报警参数

- 命令 ID：getLeakageCurrent
- 下行数据格式：

```
{
  "context":{
  "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
```

```
  "detail":""
 },
 "data":{
  "leak":[{
   "id":1, // 电流检测回路序号，int
   "switch_id":1, // 该回路对应的接触器序号，int
   "baseline":20, // 基线（屏蔽值），毫安，int，小于等于基线值的数据忽略
   "alarm_limit_1":300, // 一级漏电报警值，毫安，int：设置为 0 时表示不使用漏电报警功能
   "alarm_limit_2":300, // 二级漏电报警值，毫安，int：设置为 0 时表示不使用漏电报警功能
   "alarm_limit_3":300, // 三级漏电报警值，毫安，int：设置为 0 时表示不使用漏电报警功能
   "break_limit":500, // 漏电分匝值，毫安，int：设置为 0 时表示不使用漏电分匝功能
   "ratio": 100, // 互感比，int, 100/5 填 100,其他依次类推
   "work_model":1 // 工作模式，int,0-不启用，1-只检测，2-检测并报警，3-检测不报警但分闸，4-检测既报警又分闸
  },...]
  }
}
```

3.4.5、读取设备实时数据

支持模组功能 1 的设备，最大应支持 8 路输出控制以及 48 路电流采样功能

- 命令 ID：getRuntime
- 下行数据格式：

```
{
  "context":{
  "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{
    "voltage_a":230.45, // a 相电压，float
    "voltage_b":230.45, // b 相电压，float
    "voltage_c":230.45, // c 相电压，float
    "temperature":45.9, // 摄氏温度，float
    "loop":[{
```

"id":1, // 回路序号，int，从 1 开始

"phase":1, // 该回路对应的相位，int：1-A 相，2-B 相，3-C 相

"voltage":230.23, // 回路电压，V，float

"current":12.34, // 回路电流，A，float

"active_power":123.45, // 回路有功功率，W，float

"reactive_power":123.45, // 回路无功功率，W，float

"apparent_power":123.45, // 回路视载功率，W，float

"power_factor":0.89, // 功率因数，float

"total_energy":123.45, // 累计电量，kW/h，float

"st":1 // 回路通断状态，0-断/开门，100-通/关门，10～99-调光比例(仅调光模组有)

"leakage":0, // 灯具漏电,int, 0-正常,1-漏电,2-分闸(仅调光模组有)

"fault":0, // 故障状态,int 0-正常，1-光源故障，2-补偿电容故障，3-意外灭灯，4-意外亮灯，5-自熄灯，6-电压过大，7-电压过小，8-电流过大，9-电流过小

"working_on":0, // 工作状态,int 0-正常亮灯，1-调档节能，2-调光节能，3-正常关灯(仅调光模组有)

"voltage_st":0, // 电压状态,int，0-正常，1-过大，2-过小

"current_st":0 // 电流状态,int，0-正常，1-过大，2-过小

"leakage_current":23.23, //回路漏电流，mA，float

"ctl_adaptive": 1, // 节能方式，1-自适应，0-非自适应

},...],

"switch":[{

"id":1, // 接触器序号，int，从 1 开始

"st":1, // 接触器状态，int，0-关灯，1-开灯

"reason": 1, // 接触器当前状态的原因，0-上电初始，1-手动操

作，2-自动操作（时间方案），3-经纬度控制，4-光控，9-漏电断开（仅适用于`st==0`时）

```
    },...]
  }
}
```

3.4.6、读取硬件信息

- 命令 ID：getInfo
- 下行数据格式：

```
{
  "context":{
  "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
```

```
    "detail":""
  },
  "data":{
    "imei":"12345678900987654321", // 设备 imei，string
    "imsi":"12345678900987654321", // 设备 imsi，string
    "iccid":"12345678900987654321", // 设备 iccid，string
    "software":"abc123345", // 设备软件版本信息，string
    "hardware":"1234v323" // 设备硬件版本，string
  }
}
```

3.4.7、读取设备定位信息

- 命令 ID：getLocation
- 下行数据格式：

```
{
  "context":{
  "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
```

```
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":"",
  },
  "data":{
    "gps_status":1, // 定位状态，0-未定位，1-已定位
    "gps_type":1, // 定位类型，1-gps 定位，2-北斗定位，3-混合定位
    "lng":123.45654648, // 设备位置精度，float, wgs84 坐标系
    "lat":12.32542254 // 设备位置纬度，float, wgs84 坐标系
  }
}
```

3.4.8、读取回路参数

- 命令 ID：getLoopParam
- 下行数据格式：

```
{
  "context":{
  "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "energy":{ // 电能参数，无电能功能的设备可以不填
      "ratio_a":50, // a 相电能互感比，50:5 填 50，int
      "ratio_b":50, // b 相电能互感比，50:5 填 50，int
      "ratio_c":50, // c 相电能互感比，50:5 填 50，int
    },
    "voltage":{ // 电压限值参数
      "up_limit":250.00, // 过压报警值，float
      "up_limit_dispel":220.00, // 过压报警消除值，float
      "low_limit":200.00, // 欠压报警值，float
      "low_limit_dispel":220.00 // 欠压报警消除值，float
      },
      "loop":[{ // 回路参数以及限值
      "loop_id":1, // 电流检测回路序号，从 1 开始，int
      "switch_id":1, // 该回路对应的输出接触器序号，从 1 开始，0-表示不对应输出接触器，int
      "phase":1, // 该回路对应的相位，1-A 相，2-B 相，3-C 相，int
      "hopping":1, // 回路触点状态，1-常开，0-常闭，int
```

```
      "ratio",50, // 回路变比，50:5 填 50，int
      "levels":[{ // 多个时间段设置不同时，提交多段
        "dt_start":"20:00", // 报警起始时间，24 小时制，string，全天有效时，设置为`00:00`
        "dt_end":"22:00", // 报警结束时间，24 小时制，string，全天有效时，设置为`24:00`
        "up_limit":23.4, // 过流报警值，float
        "up_limit_dispel":20.00, // 过流报警消除值，float
        "low_limit":4.1, // 欠流报警值，float
        "low_limit_dispel":8.00 // 欠流报警消除值，float
      }...],
      },...]
  }
}
```

3.4.9、读取电能数据

- 命令 ID：getEnergy
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "active_energy_a":123.45, // a 相有功电能，kW/h，float
    "active_energy_b":123.45, // b 相有功电能，kW/h，float
    "active_energy_c":123.45, // c 相有功电能，kW/h，float
    "reactive_energy_a":123.45, // a 相无功电能，kW/h，float
    "reactive_energy_b":123.45, // b 相无功电能，kW/h，float
    "reactive_energy_c":123.45, // c 相无功电能，kW/h，float
    "total_energy":567.98, // 累计电能，kW/h，float
  }
}
```

3.4.10、读取运行参数

- 命令 ID：getRunningParam
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
```

```
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID
  },
  "data":{
    "time_ticker":{
      "heartbeat":60, // 心跳周期，单位秒，int，0-不主动上报，默认 60
      "data":1800, // 数据主动上报周期，单位秒，int，0-不主动上报，默认 1800
      "lux":15 // 光照度数据推送周期，单位秒，int,0-不主动上报，默认 15
    },
    "coord":{ // 经纬度数据，wgs84 数据
      "lng":123.32, // 经度，float
      "lat":32.32 // 纬度，float
    },
```

```
    "others":{
      "max_switch":1 // 最大控制数量
    }
  }
}
```

3.4.11、读取调光参数

- 命令 ID：getDimmingParam
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID
  },
```

```
  "data":{
    "dimming":[{
      "switch_id":1, // 回路接触器序号，int，从 1 开始
      "power_on_default":0, // 默认上电动作状态，int, 1-上电开灯，
0-上电关灯
      "rated_power":250, // 灯具额定功率，int，单位 W，0-不设置
      "dimming_type":0, // 调光方式，int，0-正向调光，1-反向调光
    },...]
  }
}
```

3.4.12、读取漏电数据

- 命令 ID：getLeakageData
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{}
}
```

- 推送数据

```
{
  "context":{
    "dev_attr":3, // 设备模组,
```

```
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "hardware_type":1, // 是否发生复位，int, 0-未发生，1-发生
    "hardware_reset":1, // 设备类型,int,0-不支持漏电，1-支持漏电
    "elu_status":1, // 状态,int, 0-未启用漏电，1-启用漏电
    "eeprom":0, // 参数状态，int,0-正常，1-错误（需要重新下发参数）
    "loop":[{
      "switchout_manually":0, // 开关量输出是否手动状态,int，0-自
动，1-手动
      "switchout_status":1, // 开关量输出当前状态，0-断开，1-闭合
      "elu_mark":0, // 功能标示,int,0-不启用,1-只检测漏电流，2-检
测漏电流并报警，3-检测漏电流不报警但分闸，4-检测漏电流报警并分闸
      "alarm_status":0, // 报警状态,int,0:未报警；1：一级报警；2：
二级报警...；0xAA：分闸报警
      "current_now": 123, // 当前值漏电流值（mA）,int
      "current_alarm":233, // 报警值（mA）,int
      "baseline":0 // 基线（屏蔽值）（mA）,int
    },...]
  }
}
```

3.4.13、读取电磁阀门开度

- 命令 ID：getSolenoidValveDegree
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "addr":41 // 阀门 modbus 地址
  }
}
```

- 推送数据

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "addr":41, // 阀门 modbus 地址
    "degree1":10, // 阀门开度,int,0-100
    "degree2":10, // 阀门开度,int,0-100
    "degree3":10, // 阀门开度,int,0-100
    "degree4":10, // 阀门开度,int,0-100
```

```
  }
}
```

3.5、控制类指令

3.5.1、复位

设备应该先应答，再进行复位动作

- 命令 ID：reset
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "do":1, // 复位类型，int，1-模块重连，2-通信参数恢复出厂并重连，3-复位设备，但保留参数，4-复位设备，将参数恢复到出厂值,5-电量等累计数据清零
  }
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000002011", // 设备标识,
```

```
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{}
}
```

3.5.2、开关灯/调光实时控制

- 命令 ID：ctlOnOff
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "switch":[{
      "id":1, // 接触器序号，int
      "opt":100, // 操作内容，int：0-关灯，100-开灯，10～99-调光比例
    },...]
  }
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":""
  },
  "data":{
    "switch":[{
      "id":1, // 接触器序号，int
      "opt":100, // 接触器当前状态，int：0-关灯，100-开灯，10～
99-调光比例，255-未知
      "st":1, // 0-操作失败，1-操作成功
    },...]
      }
}
```

3.5.3、漏电消除

- 命令 ID：leakageRecover
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
```

```
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "loop":[{
      "id":1, // 操作的回路，int，从 1 开始
      "do":1, // 消除内容，int，1-消除漏电报警，2-消除漏电分匝
（消除后，可执行开关灯指令）
    },...]
  }
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":"",
  },
  "data":{}
}
```

3.5.4、设置设备停运/投运

- 命令 ID：workingModule

- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "do":1, // 工作模式，0-停运，1-投运
  }
}
```

- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"01000000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":"",
  },
  "data":{}
}
```

3.5.5、电磁阀阀度控制

- 命令 ID：ctlSolenoidValveDegree
- 下行数据格式：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
  },
  "data":{
    "addr":41, // 阀门 modbus 地址
    "degree1":10, // 电磁阀门 1 角度，int, 0-关，100-开，0~100-自定义角度，360-不操作
    "degree2":10, // 电磁阀门 2 角度，int, 0-关，100-开，0~100-自定义角度，360-不操作
    "degree3":10, // 电磁阀门 3 角度，int, 0-关，100-开，0~100-自定义角度，360-不操作
    "degree4":10, // 电磁阀门 4 角度，int, 0-关，100-开，0~100-自定义角度，360-不操作
  }
}
```

- 推送数据

```
{
  "context":{
    "dev_attr":3, // 设备模组,
```

```
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":"",
  },
  "data":{
    "addr":41, // 阀门 modbus 地址
    "valve_id": 1, // 阀门 id,int,1-4
    "st":1 // 控制状态，0-失败，1-成功
  }
}
```

3.6、设备主动推送类协议（仅上行有）

3.6.1、设备登陆/心跳

当设备成功连接服务器时，或者经过指定的时间周期时，均应推送该消息

- 命令 ID：heartbeat
- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
```

```
    "detail":"",
  },
  "data":{
    "net_type":1, // 网络类型，int：1-窄带物联网，2/3/4/5-2G/3G/4G/5G
    "signal":21, // 信号强度，int：1～31,GPRS/4G 信号强度 CSQ 值
    "remote":"200.151.20.47:3020", // 设备的远端 ip 和端口，string
    "time":123456789, // 设备时钟时间，unix 时间戳，精确到秒，int64
  }
}
```

3.6.2、定时数据推送

- 命令 ID：postRuntime
- 推送数据：

  同 3.4.7 读取设备实时数据 上行格式

3.6.3、报警推送

- 命令 ID：postAlarm
- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":"",
```

```
  },
  "data":{
    "alarm":[{
      "loop_id":1, // 回路序号，int，从 1 开始
      "alarm_id":1, // 故障 id，int
      "phase":1, // 该回路对应的相位，故障 4,5 时需要填充，int：1-A 相，2-B 相，3-C 相
      "alarm_ts":1688716400624, // 报警/恢复时间戳，unix 格式，精确到毫秒，int64
      "alarm_value":123.123, // 报警/恢复时的数据，float：如漏电流，过流电流值等
      "alarm_level":1, // 报警/恢复的等级，漏电流，电流限值相关报警有。
      "voltage":123.12, // 发生时电压，float，V
      "current":123.12, // 发生时电流，float, A
      "active_power":123.12, // 发生时有功功率，float, kw
      "reactive_power":123.12, // 发生时无功功率，float, kw
      "power_factor":0.91, // 发生时功率因数，float,
      "switch_st":0, // 发生时控制状态，int, 0-关灯，1-开灯，2-调光
      "st":1, // 报警状态，int：1-报警，2-恢复
    },...]
    }
}
```

- 故障 ID 对照表

3.6.4、光照度数据推送

未设置 3.3.6 协议时，默认按照 15 秒间隔上送

- 命令 ID：postLux
- 推送数据：

```
{
  "context":{
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":"",
  },
  "data":{
    "lux_value":123.45, // 光照度值，float
    "lux_error":0, // 0-设备无故障，1-设备有故障，int
      }
}
```

3.7、远程升级类协议

3.7.1、设置升级用 ftp 服务参数

- 命令 ID：ftpUpgrade
- 下行数据格式：

```
{
  "context":{ // 除时间戳字段外，其他内容应与下行指令一致
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
```

```
    "serial_id":"a001", // 指令标识 ID,
  },

  "data":{
    "ftp_ip":"192.168.1.56", // 升级用 ftp 服务 ip 地址
    "ftp_port":1234, // 升级用 ftp 服务端口号
    "ftp_user":"abcd", // ftp 登录用户名
    "ftp_pwd":"1234", // ftp 登录密码
    "file_path":"/", // 升级文件存放路径
    "file_name":"v1-2-3.bin", // 升级用文件名
  }
}
```

- 推送数据 1：（指令应答）

```
{
  "context":{ // 除时间戳字段外，其他内容应与下行指令一致
    "dev_attr":3, // 设备模组,
    "dev_id":"010000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":"",
  },
  "data":{}
}
```

- 推送数据 2：（升级完成后推送）

```
{
  "context":{ // 除时间戳字段外，其他内容应与下行指令一致
  "dev_attr":3, // 设备模组,
    "dev_id":"01000000000000002011", // 设备标识,
    "cmd":"anyCmd", // 命令 id,
    "unix_milli":1722994330451, // 毫秒级 Unix 时间戳,
    "serial_id":"a001", // 指令标识 ID,
    "status":1,
    "detail":"",
  },
  "data":{
    "ver_now":"1234", // 升级后软件版本
    "ver_date":"2023-01-01", // 升级软件发布日期
    "upg_start":123456789, // 升级开始时间，unix 时间戳，精确到秒
    "upg_finish":123456789, // 升级完成时间，unix 时间戳，精确到秒
  }
}
```

4、附录

4.1、故障 ID 对照表

非回路故障可将 loop_id 填充为 0

故障 ID	故障名称	是否回路故障	说明
1	异常亮灯	Y	关灯时段内接触器吸合且有电流
2	异常灭灯	Y	亮灯时段内接触器断开且无电流
3	设备断电	N	
4	过压	N	需填充 phase
5	欠压	N	需填充 phase
6	过流	Y	
7	欠流	Y	
8	漏电报警	Y	
9	漏电分匝	Y	
10	欠压保护断电	N	需填充 phase
11	过压保护断电	N	需填充 phase
12	融断器开路	Y	亮灯时段内接触器吸合且无电流
13	缺相报警	N	需填充 phase
14	接触器吸合	Y	关灯时段内接触器吸合且无电流
15	白天有电流	Y	关灯时段内接触器未吸合且有电流
16	自熄灯/频闪	Y	
17	光源故障	Y	
18	功率因数低	Y	