Message Queuing Telemetry Transport（MQTT）

|  |  |  |  |
| --- | --- | --- | --- |
| **版本号** | **修订日期** | **修订内容** | **说明** |
| V1.1 | 2015/9/8 | 草稿 |  |
| V2.0 | 2016/4/8 | 重构 | Liuyuan |
| V2.1 | 2016/4/19 | 丰富报文格式，业务流程 | Leihong |
| V2.2 | 2016/7/13 | 增加设备间订阅，创建topic功能 | Leihong |
| V2.3 | 2016/11/2 | 加入disconnect | Leihong |
| V2.4 | 2016/12/7 | 支持will特性，离线topic消息，数据点订阅 | Leihong |
| V2.5 | 2016/12/26 | 支持qos2、retain特性，支持多种数据点类型上传 | Wangxj |
| V2.6 | 2017/3/13 | 离线topic消息功能限制 | Wangxj |

目录

[1 说明 4](#_Toc470269059)

[2 接入流程 4](#_Toc470269060)

[3 Packet格式说明 4](#_Toc470269061)

[3.1 Fixed header 4](#_Toc470269062)

[3.2 Variable Header &Payload 5](#_Toc470269063)

[4 支持的packet 5](#_Toc470269064)

[4.1 CONNECT 5](#_Toc470269065)

[4.1.1 Fixed Header 5](#_Toc470269066)

[4.1.2 VariableHeader 6](#_Toc470269067)

[4.1.3 Payload 6](#_Toc470269068)

[4.2 CONNACK 7](#_Toc470269069)

[4.2.1 Fixed Header 7](#_Toc470269070)

[4.2.2 VariableHeader 7](#_Toc470269071)

[4.3 PUBLISH（client -> server） 8](#_Toc470269072)

[4.3.1 Fixed header 8](#_Toc470269073)

[4.3.2 VariableHeader 8](#_Toc470269074)

[4.3.3 Payload 8](#_Toc470269075)

[4.4 PUBLISH（server -> client） 9](#_Toc470269076)

[4.4.1 Fixed header 9](#_Toc470269077)

[4.4.2 VariableHeader 9](#_Toc470269078)

[4.4.3 Payload 9](#_Toc470269079)

[4.5 PUBACK 9](#_Toc470269080)

[4.5.1 Fixed header 9](#_Toc470269081)

[4.5.2 VariableHeader 10](#_Toc470269082)

[4.6 PUBREC 10](#_Toc470269083)

[4.6.1 Fixed header 10](#_Toc470269084)

[4.6.2 VariableHeader 10](#_Toc470269085)

[4.7 PUBREL 10](#_Toc470269086)

[4.7.1 Fixed header 10](#_Toc470269087)

[4.7.2 VariableHeader 10](#_Toc470269088)

[4.8 PUBCOMP 10](#_Toc470269089)

[4.8.1 Fixed header 10](#_Toc470269090)

[4.8.2 VariableHeader 11](#_Toc470269091)

[4.9 SUBSCRIBE 11](#_Toc470269092)

[4.9.1 Fixed header 11](#_Toc470269093)

[4.9.2 VariableHeader 11](#_Toc470269094)

[4.9.3 Payload 11](#_Toc470269095)

[4.10 SUBACK 12](#_Toc470269096)

[4.10.1 Fixed header 12](#_Toc470269097)

[4.10.2 VariableHeader 12](#_Toc470269098)

[4.10.3 Payload 12](#_Toc470269099)

[4.11 UNSUBSCRIBE 12](#_Toc470269100)

[4.11.1 Fixed header 12](#_Toc470269101)

[4.11.2 VariableHeader 13](#_Toc470269102)

[4.11.3 Payload 13](#_Toc470269103)

[4.12 UNSUBACK 13](#_Toc470269104)

[4.12.1 Fixed header 13](#_Toc470269105)

[4.12.2 VariableHeader 13](#_Toc470269106)

[4.13 PING 13](#_Toc470269107)

[4.13.1 Fixed header 13](#_Toc470269108)

[4.13.2 VariableHeader 13](#_Toc470269109)

[4.14 PINGRSP 14](#_Toc470269110)

[4.14.1 Fixed header 14](#_Toc470269111)

[4.14.2 VariableHeader 14](#_Toc470269112)

[4.15 DISCONNECT 14](#_Toc470269113)

[4.15.1 Fixed header 14](#_Toc470269114)

[4.15.2 VariableHeader 14](#_Toc470269115)

[5 接入流程 15](#_Toc470269116)

[5.1 连接鉴权 15](#_Toc470269117)

[5.2 消息发布 15](#_Toc470269118)

[5.2.1 数据点上报 15](#_Toc470269119)

[5.2.2 平台命令(下发&回复) 22](#_Toc470269120)

[5.3 创建Topic 25](#_Toc470269121)

[5.4 订阅 26](#_Toc470269122)

[5.5 取消订阅 26](#_Toc470269123)

[5.6 推送设备Topic 27](#_Toc470269124)

[5.6.1 Publish报文推送: 27](#_Toc470269125)

[5.6.2 HTTP请求推送 29](#_Toc470269126)

[5.7 离线Topic 29](#_Toc470269127)

[5.8 数据点订阅 30](#_Toc470269128)

# 说明

MQTT协议详细内容请参见MQTT version 3.1.1官方文档，本文档对此不做详细说明，仅指明OneNet的要求、默认参数、以及当前实现与MQTT官方文档的差异。

该版本支持的功能:

* 鉴权;
* 数据点上报(平台指定topic);
* 创建topic;
* 获取项目的topic列表;
* 订阅/取消平台的topic;
* 设备间topic订阅;
* 平台命令下发;
* Qos0(c⬄S) ,Qos1(C->S);
* 连接保活
* 离线topic
* 数据点订阅

# 接入流程

* 1. 访问OneNET官网注册用户；
	2. 登录开发者中心，新建产品，获取项目ID、设备ID，以及authinfo等信息；
	3. 设备发送TCP连接请求地址请见文档中心FAQ，发送封装的报文与平台交互，。

# Packet格式说明

包格式包含三部分：

|  |  |
| --- | --- |
| Fixed Header | 所有packet中都必须有 |
| Varable Header | 部分包含有 |
| Payload | 部分包含有 |

## 3.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 - 5 | Remaining Length（该字段占用1-4个字节） |

该版本支持的所有类型：

|  |  |  |  |
| --- | --- | --- | --- |
| 名字 | 值 | 流向 | 描述 |
| CONNECT | 1 | C->S | 客户端请求与服务端建立连接 |
| CONNACK | 2 | S->C | 服务端确认连接建立 |
| PUBLISH | 3 | C⬄S | 发布消息 |
| PUBACK | 4 | C⬄S | 收到发布消息确认 |
| PUBREC | 5 | C⬄S | 发布消息收到 |
| PUBREL | 6 | C⬄S | 发布消息释放 |
| PUBCOMP | 7 | C⬄S | 发布消息完成 |
| SUBSCRIBE | 8 | C->S | 订阅请求  |
| SUBACK | 9 | S->C | 订阅确认 |
| UNSUBSCRIBE | 10 | C->S | 取消订阅 |
| UNSUBACK | 11 | S->C | 取消订阅确认 |
| PING | 12 | C->S | 客户端发送PING(连接保活)命令 |
| PINGRSP | 13 | S->C | PING命令回复 |
| DISCONNECT | 14 | C->S | 断开连接 |

## 3.2 Variable Header &Payload

|  |  |  |
| --- | --- | --- |
| 消息类型 | Variable Header | Payload |
| CONNECT | 有 | 有 |
| CONNACK | 有 | 有 |
| PUBLISH | 有 | 有 |
| PUBACK | 有 | 无 |
| PUBREC | 有 | 无 |
| PUBREL | 有 | 无 |
| PUBCOMP | 有 | 无 |
| SUBSCRIBE | 有 | 有 |
| SUBACK | 有 | 有 |
| UNSUBSCRIBE | 有 | 有 |
| UNSUBACK | 有 | 无 |
| PING | 无 | 无 |
| PINGRSP | 无 | 无 |
| DISCONNECT | 无 | 无 |

# 支持的packet

##  CONNECT

### 4.1.1 Fixed Header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 - 5 | Remaining Length（该字段占用1-4个字节） |

### 4.1.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1-2 | ProtocolName Length | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| byte 3 | ‘M’ | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| byte 4 | ‘Q’ | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| byte 5 | ‘T’ | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| byte 6 | ‘T’ | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| Byte7 | Protocol Level | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Byte8 | Connect Flag | Userflag | Passwordflag | WillRetainFlag | WillQos Flag | WillFlag | CleanSessionFlag | Reserve |
| Byte9-10 | KeepAlive |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

1. **版本**

必须设置为4，平台只支持版本v 3.1.1，不支持更老的版本。使用第三方客户端时需要注意选择正确的版本。

1. **user flag与password flag**

平台不允许匿名登陆，因此这两个标志位在连接时必须设置为1，否则认为协议错误，平台将会断开连接。

1. **Will flag与Willretainflag/WillQosFlag**

 **Will flag**为0时，**WillQosFlag**和**WillRetainFlag**必须为0，**Will Flag**为1时，**WillQosFlag**只能为0、1、2。

**注：**1.当该设备下一次正常上线后，服务器会清除之前的will 消息（包括retain消息）。

 2.如果设备上线时设置了Will Topic和msg，且将WillRetainFlag设置为1，只有设备异常断开后，服务器才会将该Will msg分发给新的Will Topic的订阅者，**且Qos级别以订阅的Request qos级别为准。**

3.如果设备上线时设置了Will Topic和msg，且将WillRetainFlag设置为0，服务器不会存储该消息，当设备异常断开后会将该Will msg以设备设置的WillQosFlag的Qos级别进行分发。

1. **CleanSessionFlag**

若客户端将clean session标志位设置为0，当其断开后，平台将会保存session，session需保持的内容包含：

* 客户端订阅的topic列表。
* 还未完成确认的Qos1、Qos2级别的publish消息

客户端保存session的内容包含：

* 已经发送到服务端的但还没有收到确认的Qos1、Qos2消息列表。
* 待发送的Qos0列表。

若客户端将clean session标志位设置为1，当其断开后，平台会清除设备的订阅列表及未完成确认的Qos1、Qos2的publish消息。

1. **Reserve**

保留位，置0。

1. **KeepAlive保活时间**

每个客户端可自定义设置连接保持时间，最短120秒，最长65535秒。

### 4.1.3 Payload

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Description** | **是否必须存在** | **格式** |
| Field1 | Client Identifier | 是 | 2字节字串长度 + utf8字串 |
| Field2 | UserName | 是 | 2字节字串长度 + utf8字串 |
| Field3 | UserPassword | 是 | 2字节字串长度 + utf8字串 |

与鉴权相关的字段包含client id，username和password，支持鉴权方式。

方式1：设备ID、项目ID、auth\_info

|  |  |
| --- | --- |
| 字段设置 | 消息示例 |
| client\_id设置为平台创建设备时的设备idusername设置为“项目ID”password设置为“鉴权信息（auth\_info）” | client\_id=”123”username=”433223”password=”注册的鉴权信息” |

各字段说明如下：

鉴权信息（auth\_info）：在平台申请设备时填写设备的auth\_info属性（数字+字母的字符串），该属性需要产品内具备唯一性；

方式2：设备ID + APIKey(项目ID也需要填写)

|  |  |
| --- | --- |
| 字段设置 | 消息示例 |
| client\_id设置为平台创建设备时的设备idusername设置为“项目ID” password设置为“鉴权信息（auth\_info）” | client\_id=”123”username=”433223”password=Api Key |

项目ID：在平台添加项目时平台生成的ID

APIKey：在平台上创建产品时生成的APIKey

##  CONNACK

### 4.2.1 Fixed Header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 - 5 | Remaining Length（该字段占用1-4个字节） |

### 4.2.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | Acknowledge Flags | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Sp |
| byte 2 | Return Code | x | x | x | x | x | x | x | x |

Sp: Session Present Flag，session信息在服务器已保持，置1；未保存，置0。

**返回码说明：**

|  |  |
| --- | --- |
| 返回码 | 描述 |
| 0 | 成功 |
| 1 | 协议版本错误 |
| 2 | 非法的clientid |
| 3 | 服务不可用 |
| 4 | 用户名或密码错误 |
| 5 | 非法链接(比如token非法) |

**失败：**

\*如果connect包不符合协议内容约束，则直接断掉连接，而不需要发送connack包.

 \*如果鉴权或授权失败，回复一个带非0错误码的connack包.

**成功：**

\*必须断掉重复的clientid.

\*执行cleansession 对应的操作.

\*必须回复一个connack，回复码为0.

\*开始消息传递，并加入keepalive的监视.

**PS：**客户端需要等到服务端的connack报文，才能发送后续的数据包.

## PUBLISH（client -> server）

### 4.3.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | DUP flag | QoS Level | RETAIN |
| byte2 - 5 | Remaining Length（该字段占用1-4个字节） |

 DUP:

 QoS2、Qos1：如果为0，则表示是第一次发送该包，如果为1，则表示为重复发送的包。

 Qos0：DUP必须为0

 QOS: 指定了该publish包的Qos等级如下

|  |  |  |  |
| --- | --- | --- | --- |
| Qos 值 | Bit2 | Bit1 | 描述 |
| 0 | 0 | 0 | 最多发送一次 |
| 1 | 0 | 1 | 至少发送一次 |
| 2 | 1 | 0 | 只发送一次 |

 RETAIN: 保留标志位，如果为1，服务器存储最新一条RETAIN消息，以便分发给新的订阅者，如果RETAIN为1，Payload为空，**服务器分发该消息后会清空该topic下的RETAIN消息。**

PS:该版本实现了Qos0，Qos1，Qos2。

### 4.3.2 VariableHeader

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Description** | **格式** | **是否必须** |
| Field1 | TopicName | 2字节字串长度 + utf8字串 | 是 |
| Field2 | PacketIdentifier | 2字节 | QoS0:否，QoS1、Qos2:是 |

### 4.3.3 Payload

 内容根据不同业务自定义.

## PUBLISH（server -> client）

### 4.4.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | DUP flag | Qos Level | RETAIN |
| byte2 - 5 | Remaining Length（该字段占用1-4个字节） |

 DUP:

 Qos2、QoS1：如果为0，则表示是第一次发送该包，如果为1，则表示为重复发送的包。

 Qos0：DUP必须为0

 QOS: 指定了该publish包的qos等级如下

|  |  |  |  |
| --- | --- | --- | --- |
| Qos 值 | Bit2 | Bit1 | 描述 |
| 0 | 0 | 0 | 最多发送一次 |
| 1 | 0 | 1 | 至少发送一次 |
| 2 | 1 | 0 | 只发送一次 |

 RETAIN:如果服务器分发的消息是RETAIN消息，则将该标志位置1，否则置0.

PS:该版本实现Qos0、Qos1、Qos2，其中服务器分发消息的Qos级别**是以Client->Server发送的publish包的qos等级为准，非订阅的topic的Qos等级。Retain消息是以订阅的Request Qos级别进行推送。**

### 4.4.2 VariableHeader

|  |  |  |
| --- | --- | --- |
|  | **Description** | **格式** |
| Field1 | TopicName | 2字节字串长度 + utf8字串 |

### 4.4.3 Payload

内容根据不同业务自定义.

##  PUBACK

### 4.5.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~ 5 | Remaining Length（该字段占用1-4个字节） |

### 4.5.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | PacketIdentifier | PacketIdentifier |

## PUBREC

### 4.6.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~ 5 | Remaining Length（该字段占用1-4个字节） |

### 4.6.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | PacketIdentifier | PacketIdentifier |

## PUBREL

### 4.7.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~ 5 | Remaining Length（该字段占用1-4个字节） |

### 4.7.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | PacketIdentifier | PacketIdentifier |

## PUBCOMP

### 4.8.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~ 5 | Remaining Length（该字段占用1-4个字节） |

### 4.8.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | PacketIdentifier | PacketIdentifier |

##  SUBSCRIBE

### 4.9.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~5 | Remaining Length（该字段占用1-4个字节） |

### 4.9.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | PacketIdentifier | PacketIdentifier |

### 4.9.3 Payload

|  |  |  |
| --- | --- | --- |
|  | Description | **格式** |
| Byte1~n | TopicName | 2字节字串长度 + utf8字串 |
| Byten+1 | Request Qos | 服务质量要求(只能为0、1、2) |

**topic说明**

 可以包含一个或多个topic.

topic必须是数字、英文、反斜杠（/）的组合，目前不支持通配符。

每个客户端最多订阅100个topic；

以下topic被系统保留使用，（客户端不能订阅）：

|  |  |
| --- | --- |
| 类型 | 说明 |
| $开头 | 平台保留 |

## SUBACK

### 4.10.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~ 5 | Remaining Length（该字段占用1-4个字节） |

### 4.10.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | PacketIdentifier | PacketIdentifier |

### 4.10.3 Payload

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | retcode |  |

**返回码说明：**

|  |  |
| --- | --- |
| 返回码 | 描述 |
| 0x00 | 成功(granted qos = 0) |
| 0x01 | 成功(granted qos = 1) |
| 0x02 | 成功(granted qos = 2) |
| 0x80 | 失败 |

## UNSUBSCRIBE

### 4.11.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~ 5 | Remaining Length（该字段占用1-4个字节） |

### 4.11.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | PacketIdentifier | PacketIdentifier |

### 4.11.3 Payload

|  |  |  |
| --- | --- | --- |
|  | **Description** | **格式** |
| Byte1~n | TopicName | 2字节字串长度 + utf8字串 |

 可以包含一个或多个topic.

## 4.12 UNSUBACK

### 4.12.1 Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~ 5 | Remaining Length（该字段占用1-4个字节） |

### 4.12.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | PacketIdentifier | PacketIdentifier |

## PING

### Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~ 5 | Remaining Length（该字段占用1-4个字节） |

### 4.13.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | NULL | NULL |

注：PING命令无VariableHeader

## PINGRSP

### Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~ 5 | Remaining Length（该字段占用1-4个字节） |

### 4.14.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | NULL | NULL |

注：PINGRSP无VariableHeader

## DISCONNECT

### Fixed header

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1 | MQTT Packet Type | 0 | 0 | 0 | 0 |
| byte2 ~ 5 | Remaining Length（该字段占用1-4个字节） |

### 4.15.2 VariableHeader

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| byte 1~2 | NULL | NULL |

注：DISCONNECT无VariableHeader

# 接入流程

## 连接鉴权



* 设备向平台发起connect请求.connect中携带鉴权信息,具体参见(报文格式参考4.1)
* 平台拿到鉴权信息进行鉴权.
* 鉴权通过后，如果cleansession=0, 平台将会加载保存的设备的一些信息.如订阅列表(4.1.2中描述).

如果cleansession=1, 设备没有保存信息在平台，则不加载设备相关信息.

* 返回鉴权结果ConnAck(报文格式参考4.2).

## 消息发布

### 5.2.1 数据点上报

设备使用publish报文来上传数据点， 报文格式如下：

 VariableHeader：

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Field名称** | **说明** | **格式** |
| Field1 | TopicName=”**$dp**” | $dp为系统上传数据点的指令 | 2字节字串长度 + utf8字串  |

Payload:

Payload包含真正的数据点内容,支持的格式如下：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **字节** | **说明\bit** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 1 | Bit0-5数据类型指示，目前支持： Type = 1 Type = 2 Type = 3Type = 4Type = 5Type = 6Type = 7Bit6-7 flags 根据具体类型不同意义Type6、Type7这两种需要填写时间戳的数据类型：如果填入了时间戳，**则须将bit7置1**，否则置0(即忽略数据类型说明中的日期、时间相关字段) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Byte 2 | 根据数据类型指示不同 |  |  |  |  |  |  |  |  |
| 。。。 |  |  |  |  |  |  |  |  |
| Byte n |  |  |  |  |  |  |  |  |

数据类型1(type == 1)格式说明：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 1 | 数据点类型值：1 //1：JSON格式1字符串 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Byte 2 | *//指示后面json字符串长度*固定两字节长度高位字节，值为0x00 |  |  |  |  |  |  |  |  |
| Byte 3 | 固定两字节长度低位字节，值为0x41 |  |  |  |  |  |  |  |  |
| Byte 4 | { “datastreams”:[*// 可以同时传递多个数据流* { “id”:”temperature”, “datapoints”:[//数据流可以有多个数据点{”at”:”2013-04-22 22:22:22”,*//可选*”value”: 36.5*//用户自定义*}]},{ “id”:”location”“datapoints”:[…]}, { … }]} |  |  |  |  |  |  |  |  |
| ………………………… |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| Byte n |  |  |  |  |  |  |  |  |

数据类型2(type == 2)格式说明：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 1 | 数据点类型指示：type=2 //二进制数据 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Byte 2 | *//指示后面json字符串长度*固定两字节长度-高位字节，值为0x00 |  |  |  |  |  |  |  |  |
| Byte 3 | 固定两字节长度-低位字节，值为0x10 |  |  |  |  |  |  |  |  |
| Byte 4 | {”ds\_id”:”image”, *//创建数据流时定义的ID，（必填）*“at”:”2014-10-25 12:23:23”, *//时间，（可选）*”desc”:字符串或json对象*//对该数据的描述（可选）*} |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |  |
| Byte n |  |  |  |  |  |  |  |  |
| Byte n+1 | *//指示后面二进制数据长度*固定四字节长度-第1字节(最高)，值为0x00 |  |  |  |  |  |  |  |  |
| Byte n+1 | 固定四字节长度-第2字节，值为0x00 |  |  |  |  |  |  |  |  |
| Byte n+2 | 固定四字节长度-第3字节，值为0x01 |  |  |  |  |  |  |  |  |
| Byte n+3 | 固定四字节长度-第4字节(最低)，值为0x00 |  |  |  |  |  |  |  |  |
| Byte n+4 | *//该域目前最大支持3M*本例中的该域256字节数据 |  |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |  |
| Byte n+260 |  |  |  |  |  |  |  |  |

数据类型3(type == 3)格式说明：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 1 | 数据点类型指示：type=3 *// JSON格式2字符串* | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| Byte 2 | *//指示后面字符串长度*固定两字节长度高位字节，值为0x00 |  |  |  |  |  |  |  |  |
| Byte 3 | 固定两字节长度低位字节，值为0x46 |  |  |  |  |  |  |  |  |
| Byte 4 | 通用格式：{ “datastream\_id1”:”value1”, “datastream\_id2”:”value2”,…}示例：{“temperature”:22.5,”humidity”:”95.2%”} |  |  |  |  |  |  |  |  |
| ………… |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| Byte n |  |  |  |  |  |  |  |  |

数据类型4(type ==4)格式说明

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 1 | 数据点类型指示：type=4 *//* *JSON格式3字符串* | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Byte 2 | *//指示后面字符串长度*固定两字节长度高位字节，值为0x00 |  |  |  |  |  |  |  |  |
| Byte 3 | 固定两字节长度低位字节，值为0x46 |  |  |  |  |  |  |  |  |
| Byte 4 | 通用格式：{ “datastream\_id1”:{“datetime1”:”value1”},“datastream\_id2”: {“datetime2”:”value2”},…}示例：{“temperature”:{“2015-03-22 22:31:12”:22.5}} |  |  |  |  |  |  |  |  |
| ………… |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| Byte n |  |  |  |  |  |  |  |  |

数据类型5(type == 5)格式说明

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 1 | 数据点类型指示：type=5 *//自定义分隔符* | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| Byte 2 | *//指示后面字符串长度*固定两字节长度高位字节，值为0x00 |  |  |  |  |  |  |  |  |
| Byte 3 | 固定两字节长度低位字节，值为0x41 |  |  |  |  |  |  |  |  |
| Byte 4 | 消息中最前面两字节为用户自定义的域中分隔符和域间分隔符，这两个分隔符不能相同。比如采用逗号作为域中分隔符，分号作为域间分隔符的格式如下：,;feild0;feild1;…;feildn其中，每个field格式支持3种：field格式1： 3个子字段，分别是数据流ID,时间戳，数据值。通用格式：Datastream\_id,datetime,valuefield格式2： 2个子字段，分别是数据流ID和数据值，省略时间戳。通用格式：Datastream\_id,valuefield格式3： 1个子字段，省略了数据ID和时间戳，只传输数据值，平台将用该域(feild)所在的位置号（从0开始）作为数据流ID。通用格式：value示例：(1),;temperature,2015-03-22 22:31:12,22.5;102;pm2.5,89;10(2)#@temperature#2015-03-22 22:31:12#22.5@102@pm2.5#89@10 |  |  |  |  |  |  |  |  |
| ………………………… |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| Byte n |  |  |  |  |  |  |  |  |

数据类型6(type == 6)格式说明

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 1 | 数据类型指示：type=6 *//带时间自定义分隔符* | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| Byte 2 | 年（后两位），例如2016年，则该字节为16 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Byte 3 | 月（1-12） |  |  |  |  |  |  |  |  |
| Byte 4 | 日（1-31） |  |  |  |  |  |  |  |  |
| Byte 5 | 小时（0~23） |  |  |  |  |  |  |  |  |
| Byte 6 | 分钟（0~59） |  |  |  |  |  |  |  |  |
| Byte 7 | 秒（0~59） |  |  |  |  |  |  |  |  |
| Byte 8 | *//指示后面字符串长度*固定两字节长度高位字节，值为0x00 |  |  |  |  |  |  |  |  |
| Byte 9 | 固定两字节长度低位字节，值为0x41 |  |  |  |  |  |  |  |  |
| Byte 10 | 消息中最前面两字节为用户自定义的域中分隔符和域间分隔符。具体格式见type=5 说明，若相关域中没有时间戳，则采用本类型的默认时间戳当作数据点的时间来存储。 |  |  |  |  |  |  |  |  |
| …… |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| Byte n |  |  |  |  |  |  |  |  |

数据类型7(type == 7)格式说明：（每次最多500个数据流的浮点数）

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 1 | Bit0-5数据类型指示：type=7 *//可离散浮点数数据流*Bit6：保留，置0Bit7: 时间指示位，1，携带6字节时间 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| Byte 2 | 年（后两位），例如2016年，则该字节为16 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Byte 3 | 月（1-12） |  |  |  |  |  |  |  |  |
| Byte 4 | 日（1-31） |  |  |  |  |  |  |  |  |
| Byte 5 | 小时（0~23） |  |  |  |  |  |  |  |  |
| Byte 6 | 分钟（0~59） |  |  |  |  |  |  |  |  |
| Byte7 | 秒（0~59） |  |  |  |  |  |  |  |  |
| Byte 8 | *//数据流名称（取值范围1-65535）*高位字节，值为0x00 |  |  |  |  |  |  |  |  |
| Byte 9 | 低位字节，值为0x01 |  |  |  |  |  |  |  |  |
| Byte10 | *//数据流个数（取值范围1-500）*高位字节，值为0x00 |  |  |  |  |  |  |  |  |
| Byte11 | 低位字节，值为0x01 |  |  |  |  |  |  |  |  |
| Byte 10 | 4字节float类型，低位在前，高位在后 |  |  |  |  |  |  |  |  |
| Byte 11 |  |  |  |  |  |  |  |  |
| Byte 12 |  |  |  |  |  |  |  |  |
| Byte 13 |  |  |  |  |  |  |  |  |
| ... |  |  |  |  |  |  |  |  |  |
| Byte n | *//数据流名称（取值范围1-65535）*高位字节，值为0x24 |  |  |  |  |  |  |  |  |
| Byte n+1 | 低位字节，值为0x37 |  |  |  |  |  |  |  |  |
| Byte n+2 | *//数据流个数（取值范围1-500）*高位字节，值为0x01 |  |  |  |  |  |  |  |  |
| Byte n+3 | 低位字节，值为0x00 |  |  |  |  |  |  |  |  |
| Byte n+2 | 4字节float类型，低位在前，高位在后 |  |  |  |  |  |  |  |  |
| Byte n+4 |  |  |  |  |  |  |  |  |
| Byte n+5 |  |  |  |  |  |  |  |  |
| Byte n+6 |  |  |  |  |  |  |  |  |

#### 5.2.1.1 Qos0(Client->Server)



* 设备发布Qos0消息(上报数据点)
* 平台收到上报数据点后保存起来.

#### 5.2.1.2 Qos1(Client->Server)



* 设备发布Qos1消息(上报数据点)
* 平台收到上报数据点后保存起来.
* 平台给设备回复相应的PubAck (报文格式参考4.5)

#### 5.2.1.3 Qos2(Client->Server)



* 设备发布Qos2消息(上报数据点)
* 平台收到上报数据点后保存起来
* 平台给设备回复相应的PubRec报文
* 设备需回复平台PubRel报文，如超时不回平台则会断开相应连接
* 平台给设备回复PubComp报文

注：**数据点上报功能不支持Retain特性。**

### 5.2.2 平台命令(下发&回复)

#### 5.2.2.1命令下发

平台使用publish报文来下发平台指令， 报文格式如下：

FixHeader：

参考4.3.1

 VariableHeader：

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Field名称** | **说明** | **格式** |
| Field1 | TopicName=”**$creq/cmduuid**” | $creq为系统下发Cmd的指令，cmduuid为该条指令的uuid | 2字节字串长度 + utf8字串  |

Payload:

Payload包含真正的指令内容

##### 5.2.2.1.1 Qos0(Server->Client)



* 命令下发：

平台向设备发送topic为$creq的消息(该topic为平台命令).

设备收到topic为$creq的topic时，需将其作为平台下发的指令来处理.

注：目前命令下发以Qos0级别进行推送

#### 5.2.2.2命令回复

设备使用publish报文来回复平台指令， 报文格式如下：

FixHeader：

参考4.3.1

 VariableHeader：

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Field名称** | **说明** | **格式** |
| Field1 | TopicName=”**$crsp/cmduuid**” | $crsp为系统处理设备回复cmd的指令，cmduuid为该条指令的uuid | 2字节字串长度 + utf8字串  |

Payload:

Payload包含真正回复的指令内容

##### 5.2.2.2.1 Qos0(Client->Server)



##### 5.2.2.2.2 Qos1(Client<-> Server)



 如果设备回复响应时以Qos1回复，则平台需要给设备回复一个Puback报文

##### 5.2.2.2.3 Qos2(Client<-> Server)



 如果设备回复响应时以Qos2回复，则

 1．平台需回复设备一个PubRec报文

 2．设备在收到PubRec后需向平台回复PubRel报文

 3．平台收到PubRel报文后，向设备回复PubComp报文

## 创建Topic



* 设备通过发送HTTP请求进行topic的创建操作.
* 平台收到请求后创建topic并返回结果.

请求及响应定义如下：

|  |  |
| --- | --- |
| HTTP方法 | POST |
| URL | http://api.heclouds.com/mqtt/topic |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，必须master key |
| URL参数 |  |
| HTTP body内容 | {“name”:”xxxxxx” //topic名字，只能由英文，下划线，反斜杠组成，最长1024个字符，同一产品下topic name不重复。} |
| 成功返回 | {"errno": 0,"error":“succ”} |
| 说明 | 每个产品最多100个topic |

## 订阅



* 设备发起订阅请求.(报文格式参考4.9)
* 平台收到请求后更新topic列表.
* 平台给设备回复SubAck. (报文格式参考4.10)

注：subscribe的request qos级别可以为0、1、2。

## 取消订阅



* 设备发起取消订阅请求. (报文格式参考4.11)
* 平台收到请求后更新topic列表.
* 平台给设备回复UnSubAck. (报文格式参考4.12)

## 推送设备Topic

### 5.6.1 Publish报文推送:

Qos0：设备使用Qos级别为0的Publish报文来进行推送



* 设备发起推送topic请求(以Qos0级别)
* 平台收到请求后,将topic以Qos0级别推送到相关订阅设备(支持离线设备推送)
* 平台不返回PubAck或PubRec等报文

Qos1

 设备使用Qos级别为1的publish报文来推送Topic.



* 设备发起推送topic请求(以Qos1级别)
* 平台收到请求后,将topic以Qos1级别推送到相关订阅设备(支持离线设备推送)
* 平台返回PubAck报文

Qos2

 设备使用Qos级别为2的publish报文来推送Topic.



* 设备发起推送topic请求(以Qos2级别).
* 平台收到请求后,回复PubRec报文
* 设备收到PubRec后需回复PubRel报文 (如超时不回复,平台会断开与设备的连接)
* 平台收到PubRel报文后,回复PubComp给设备。
* 平台在回复PubComp后会以Qos2级别推送到相关订阅设备(支持离线设备推送)
* 设备需回复PubRec报文 (如超时不回复,平台会断开与设备的连接)
* 平台发送PubRel报文给设备
* 设备需回复PubComp(发布完成)

 Publish报文的格式如下：

FixHeader：

参考4.3.1

 VariableHeader：

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Field名称** | **说明** | **格式** |
| Field1 | TopicName | 填写设备订阅的topic | 2字节字串长度 + utf8字串  |

Payload：

Payload为设备自定义内容

### 5.6.2 HTTP请求推送



* 设备以HTTP的方式发起推送topic请求.
* 平台收到请求后,将topic推送到相关订阅设备.(目前只支持在线推送)
* 平台返回推送结果.

请求及响应定义如下：

|  |  |
| --- | --- |
| HTTP方法 | POST |
| URL | http://api.heclouds.com/mqtt |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，必须master key |
| URL参数 | topic=“XXXXXX”，必选，在发送mqtt协议的topic. 必填 |
| HTTP body内容 | 用户自定义Json或二进制数据（小于64K） |
| 成功返回 | {"errno": 0,"error":“succ”} |

## 离线Topic

[5.6节 推送设备Topic](#_推送设备Topic)只针对在线设备进行topic消息的推送，离线设备不会收到订阅的topic的消息。离线Topic则会将该消息推送给在线以及离线设备。

注：

1. 如果设备在上线时设置了clean session flag，服务端会删除其订阅的topic及相关的离线topic消息。
2. 如果设备没有设置clean session flag，如果有与该设备相关的离线topic消息，则在鉴权成功后将离线消息推送给设备。
3. 遗嘱消息(will msg)只会推送给订阅了该will topic的**在线**的设备，离线设备不会收到。
4. **离线消息的有效期默认为2天(暂不支持用户设定有效期)，服务器只会推送在2天内的最近10条消息。**

## 数据点订阅

数据点订阅：同一产品下的设备可以订阅其他设备的数据点，订阅的topic格式为：/deviceid/数据流名称。即

被关注的设备在上传了该数据流名称的数据点后，订阅了该topic的其他设备都会收到上传的数据点。

 例：

1. A、B设备的deviceid分别为9277、9278。

2. A设备订阅了名为/9278/9527的topic(9278为设备B的id，9527为B设备上传的数据流名称)。

3. B设备上传了名为9527的数据流(数据点为11; 15;78…)。

4. A设备会收到多条(取决于设备B上传的数据点的个数)推送的topic名为/9278/9527的publish消息，消息的payload中则包含设备B上传的数据点。

注：目前支持订阅的数据点的类型为Type1~Type7(详见[5.2.1节](#_5.2.1_数据点上报))。

PS: 欢迎访问设备云门户网站<http://open.iot.10086.cn/>注册用户，获取最新文档。