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
V2.7	2017/7/12	暂时屏蔽 Retain 特性(包括 will 消息)	Wangxj
V2.8	2017/11/14	subscribe 支持通配符,支持 Retain 特性	Wangxj

目录

1 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

2 接入流程

2.1 访问平台 http://open.iot.10086.cn/注册用户；

2.2 用户根据业务情况，在“连接请求”章节中选择 EDP 登录方式（目前公测阶段，页面还未提供 MQTT 登录选项，登录方式与 EDP 兼容）；

2.3 登录需填写设备相关属性，在项目下新增设备，获取项目 ID、设备 ID，以及 authinfo 等信息；

2.4 设备发送 TCP 连接请求到以下地址，发送封装的报文与平台交互。

平台服务器地址 183.230.40.39，TCP 端口 6002

3 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

4 支持的 packet

4.1 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	0	0	0	0	0	0	0	0
	Length	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	User flag	Password flag	WillRetain Flag	WillQos Flag		WillFlag	CleanSession Flag	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。

（2） 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。

（3） 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.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，**分发的 Qos 级别为订阅的 Request qos 级别和发布 Will 消息的 Qos 级别中的较小者。**

2.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。

（4） 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。

（5） Reserve

保留位，置 0。

（6） 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 id username 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 id username 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

4.2 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.

4.3 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；或者是 RETAIN=1，Qos=0，**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

注：**发布时 TopicName 只能为字母、数字、反斜杠、下划线，如有其它非法符号，服务器会忽略该消息**

4.3.3 Payload

内容根据不同业务自定义.**Payload 长度不能超过 1MB**

4.4 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。(如果分发的是未完成确认的 Qos > 0 的消息，则 DUP flag = 1)

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:如果该 Publish 消息是回应的 subscribe 报文请求（即分发的是 RETAIN 消息），对于新的订阅，RETAIN=1，对于匹配已有的订阅则 RETAIN = 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 Request Qos 等级。**

分发的 Retain 消息的 Qos 级别为订阅的 Request Qos 级别和发布 RETAIN 消息的 Qos 级别中的较小者。

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.

4.5 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							

4.6PUBREC

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							

4.7PUBREL

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

4.8 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							

4.9 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、下划线（_）、反斜杠（/）的组合，**支持通配符(+、#)，不支持其余特殊符号**

每个客户端最多订阅 **50** 个 topic；

以$符号开头的 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

4.10 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

4.11 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							

4.13PING

4.13.1 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）							

4.13.2 VariableHeader

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

注：PING 命令无 VariableHeader

4.14PINGRSP

4.14.1 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）							

4.14.2 VariableHeader

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

注：PINGRSP 无 VariableHeader

4.15DISCONNECT

4.15.1 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）							

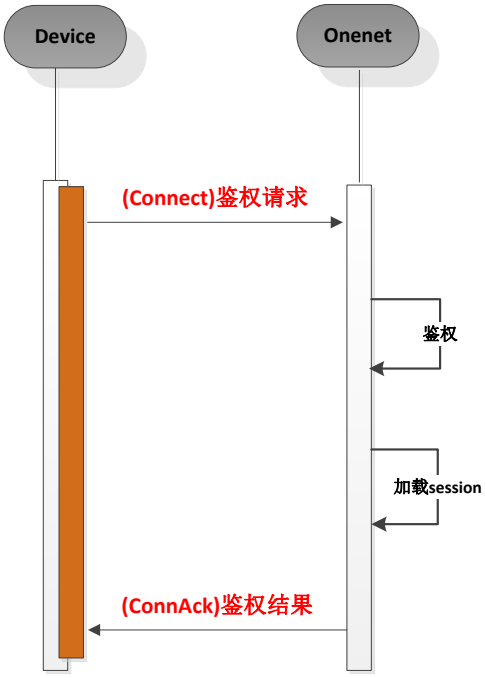
4.15.2 VariableHeader

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

注：DISCONNECT 无 VariableHeader

5 接入流程

5.1 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).

5.2 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 = 3 Type = 4 Type = 5 Type = 6 Type = 7 Bit6-7 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）*								

Byte n	”desc”:字符串或 json 对象*//对该数据的描述（可选）* }								
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”,								
…	…								
…	}								
Byte n	示例： {“temperature”:22.5,”humidity”:”95.2%”}								

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”},								
…	…								
…	}								
Byte n	示例： {“temperature”:{“2015-03-22 22:31:12”:22.5}}								

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,value field 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,value field 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

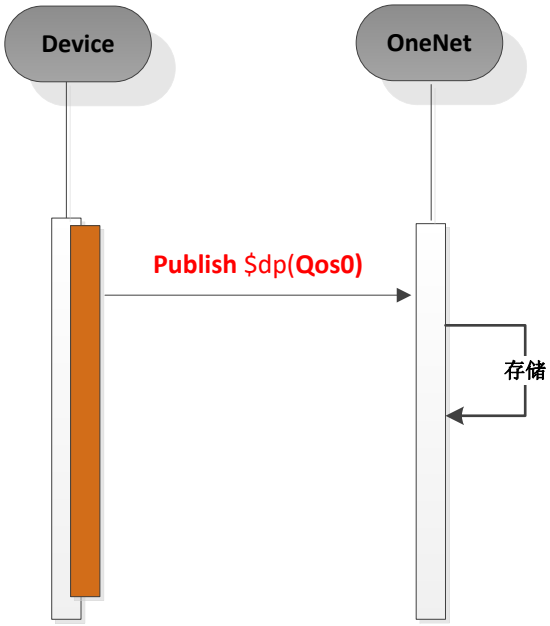
Byte 1	数据类型指示：type=6　　*//带时间自定义分隔符*	0	0	0	0	0	1	1	0
Byte 2	年（后两位），例如 2016 年，则该字节为 16	0	0	0	1	0	0	0	0
Byte 3	月（1-12）								
Byte 4	日（1-31）								
Byte 5	小时（0~23）								
Byte 6	分钟（0~59）								
Byte 7	秒（0~59）								
Byte 8	*//指示后面字符串长度* 固定两字节长度高位字节，值为 0x00								
Byte 9	固定两字节长度低位字节，值为 0x41								
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 0 Bit7: 时间指示位，1，携带 6 字节时间	1	0	0	0	0	1	1	1

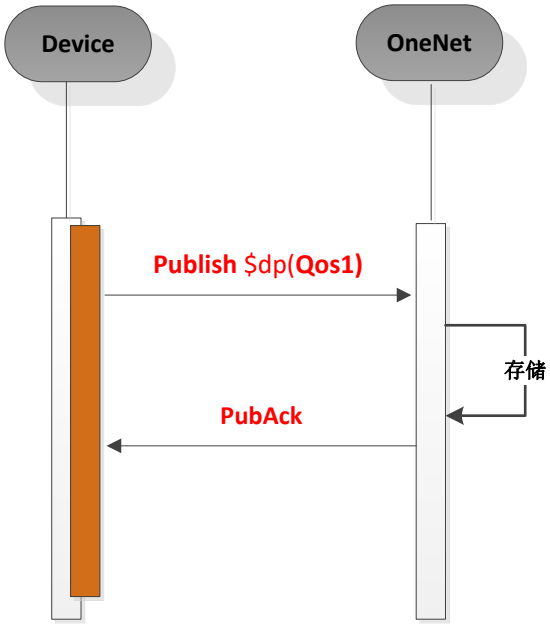
Byte 2	年（后两位），例如 2016 年，则该字节为 16	0	0	0	1	0	0	0	0
Byte 3	月（1-12）								
Byte 4	日（1-31）								
Byte 5	小时（0~23）								
Byte 6	分钟（0~59）								
Byte7	秒（0~59）								
Byte 8	*//数据流名称（取值范围 1-65535）* 高位字节，值为 0x00								
Byte 9	低位字节，值为 0x01								
Byte10	*//数据流个数（取值范围 1-500）* 高位字节，值为 0x00								
Byte11	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.

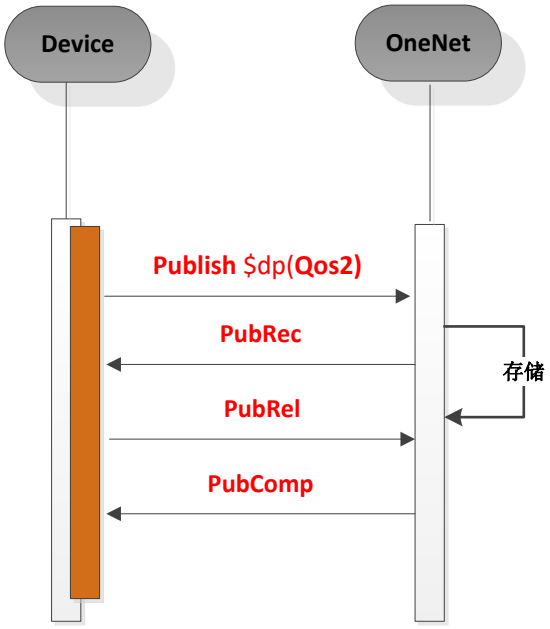
5.2.1.2 Qos1(Client->Server)



- 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：

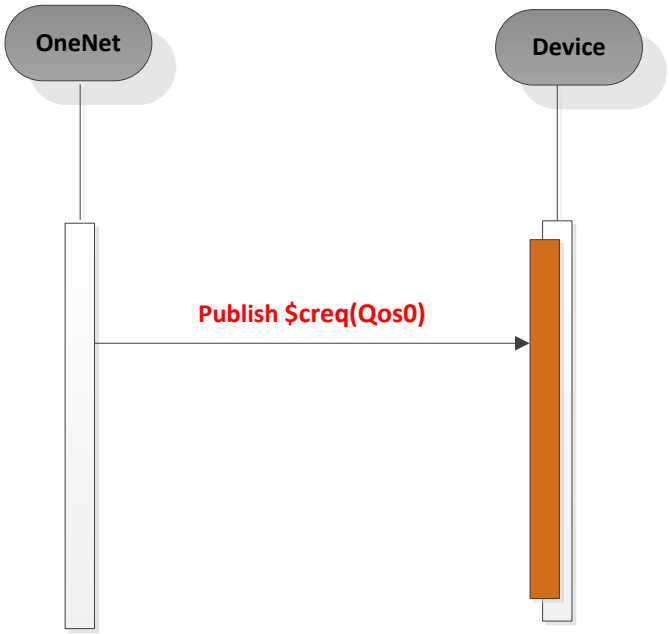
FixHeader：
参考 4.3.1

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：

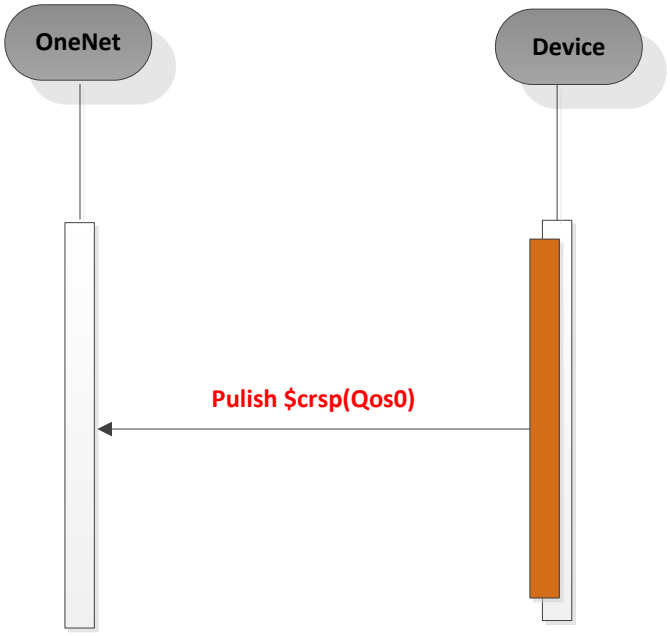
FixHeader：
参考 4.3.1

VariableHeader：

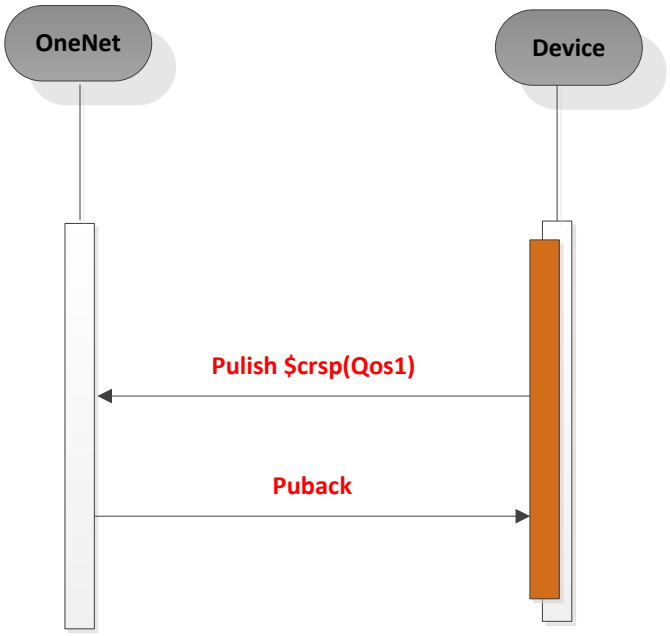
	Field 名称	**说明**	**格式**
Field1	TopicName=”`$crsp/cmduuid`”	`$crsp` 为系统处理设备回复 `cmd` 的指令，`cmduuid` 为该条指令的 `uuid`	2 字节字串长度 + utf8 字串

Payload:
Payload 包含真正回复的指令内容，**命名回复内容长度不能超过 64KB（超过 64KB 的部分会被截断）**

5.2.2.2.1 Qos0(Client->Server)

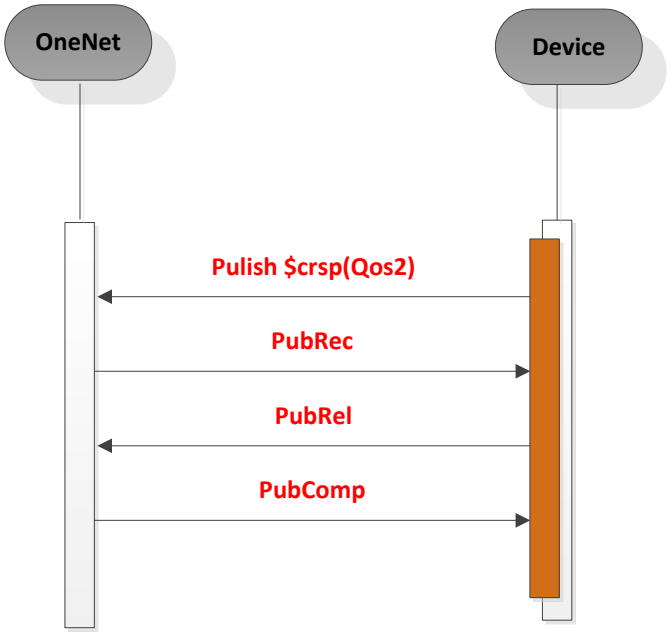


5.2.2.2.2 Qos1(Client<-> Server)



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

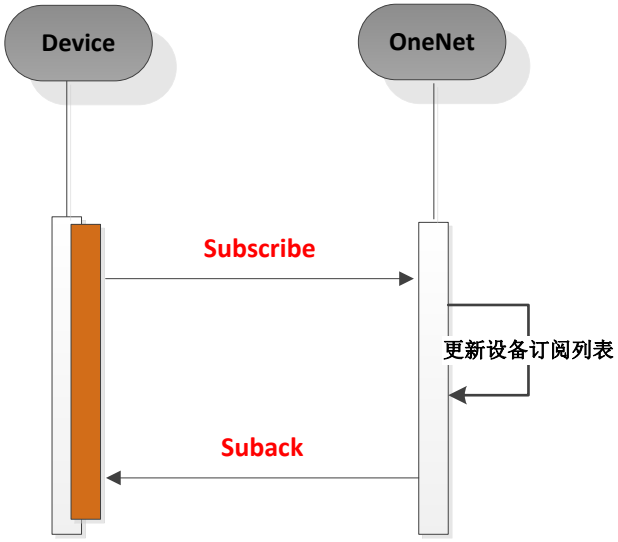
5.2.2.2.3 Qos2(Client<-> Server)



如果设备回复响应时以 Qos2 回复，则

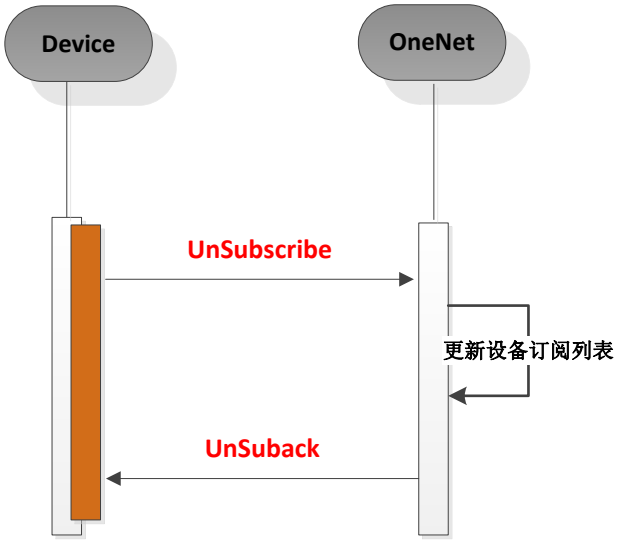
1. 平台需回复设备一个 PubRec 报文
2. 设备在收到 PubRec 后需向平台回复 PubRel 报文
3. 平台收到 PubRel 报文后，向设备回复 PubComp 报文

5.3 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。

5.4 取消订阅

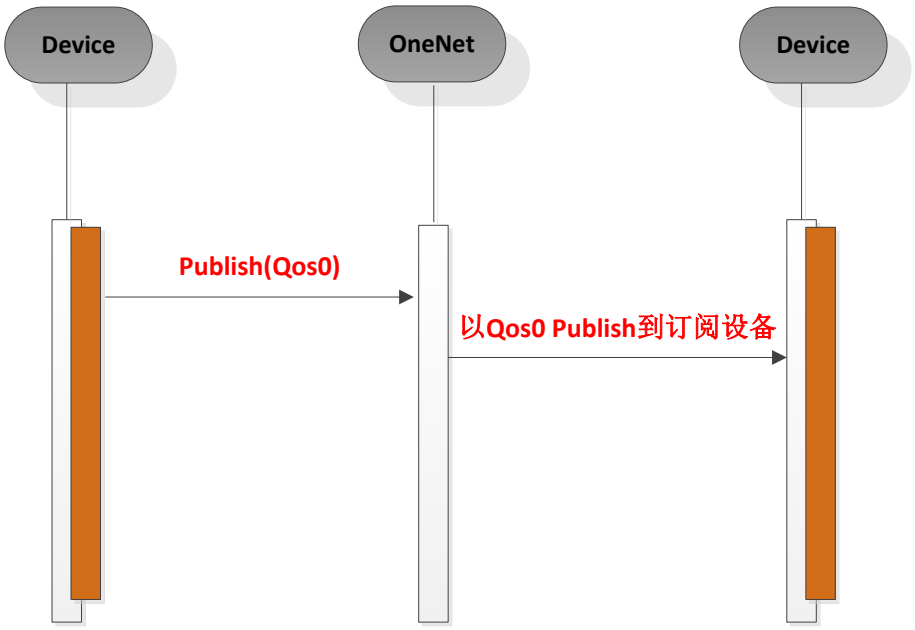


- 设备发起取消订阅请求. (报文格式参考 4.11)
- 平台收到请求后更新 topic 列表.
- 平台给设备回复 UnSubAck. (报文格式参考 4.12)

5.5 推送设备 Topic

5.5.1 Publish 报文推送:

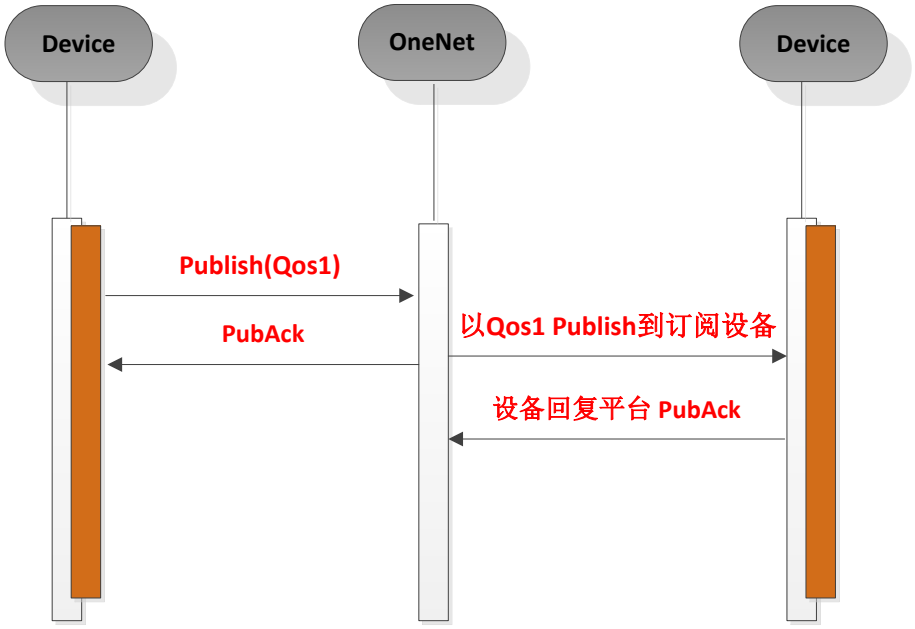
Qos0：设备使用 Qos 级别为 0 的 Publish 报文来进行推送



- 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.

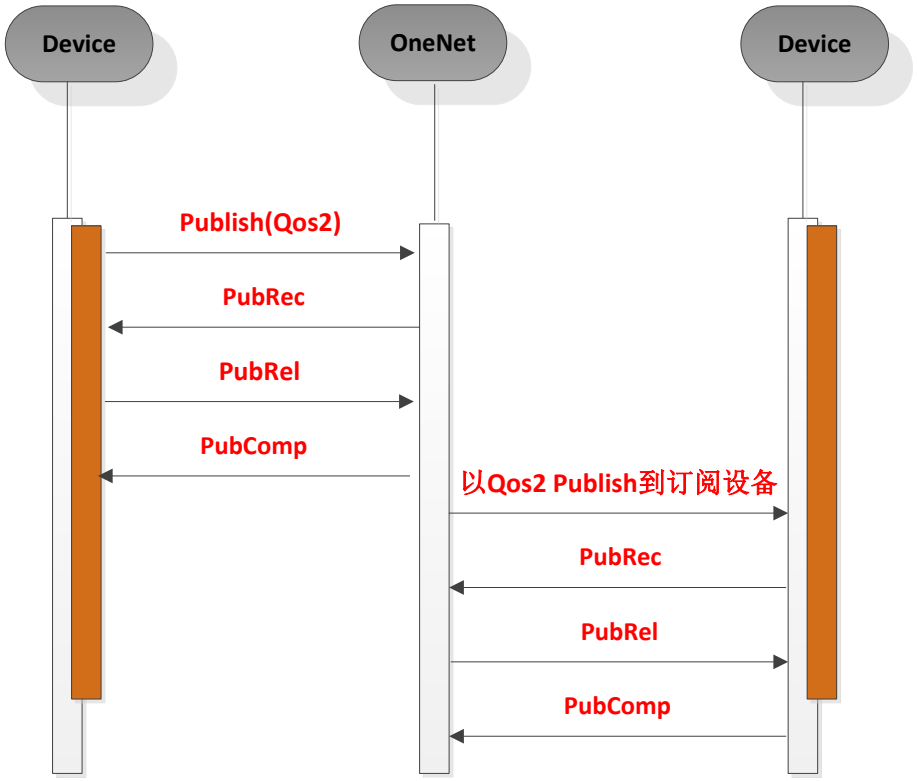


- 设备发起推送 topic 请求(以 Qos1 级别)
- 平台收到请求后,将 topic 以 Qos1 级别推送到相关订阅设备(支持离线设备推送)

- 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
- 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
- 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

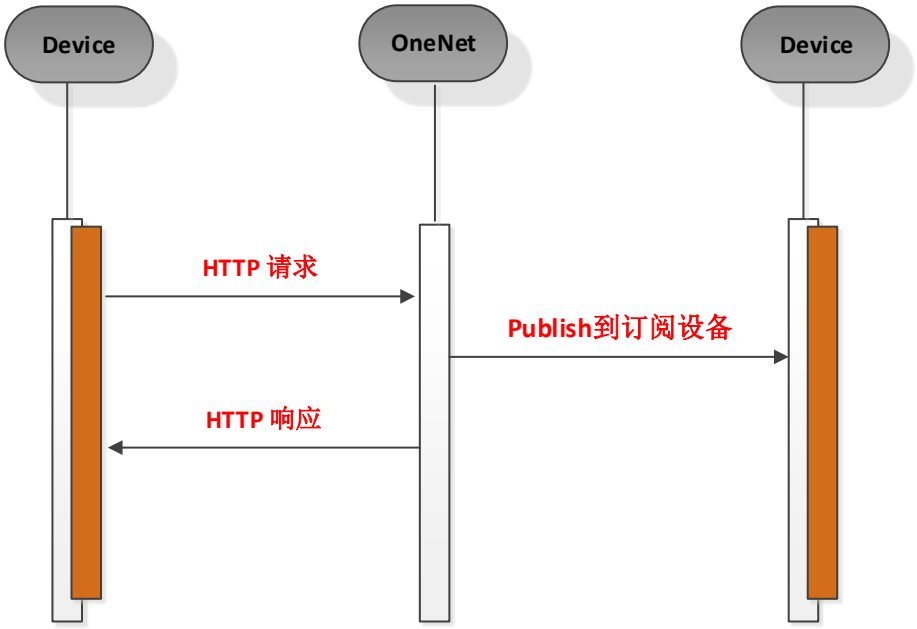
VariableHeader：

	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

5.5.2 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”
	}

5.6离线 Topic

5.5 节 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（按消息生成时间升序推送）。**

5.7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。
5. 目前支持 topic 通配符，如订阅产品下的所有设备上传的 temp 数据流的数据点，订阅的 topic 为**/+/temp**

注：1.目前支持订阅的数据点的类型为 Type1~Type7(详见 5.2.1 节)。

2.数据点订阅**只限于**设备通过 MQTT 协议上传的数据流、数据点，其他通过 HTTP 方式上传的数据点不能订阅。