**传感器网络MQTT（MQTT-SN）**

**协议文档**

**1.2版本**

**2013年11月14日**

目录

[1. 更改和修订历史 3](#_Toc17617715)

[**1.1 版本1.0，2007年11月29日** 3](#_Toc17617716)

[**1.2 版本1.1，2008年6月5日** 3](#_Toc17617717)

[**1.3 版本1.2，2011年5月20日** 3](#_Toc17617718)

[2. 简介 3](#_Toc17617719)

[3. MQTT-SN与MQTT比较 4](#_Toc17617720)

[4. MQTT-SN架构 5](#_Toc17617721)

[**4.1 透传网关** 6](#_Toc17617722)

[**4.2 聚合网关** 6](#_Toc17617723)

[5. 消息格式 6](#_Toc17617724)

[**5.1 基本消息格式** 6](#_Toc17617725)

[**5.2 消息头** 7](#_Toc17617726)

[**5.3 消息可变部分** 8](#_Toc17617727)

[**5.4 消息格式** 10](#_Toc17617728)

[**5.5 转发器封装** 19](#_Toc17617729)

[6. 功能描述 20](#_Toc17617730)

[**6.1 网关的广播和发现** 20](#_Toc17617731)

[**6.2 客户端的连接设置** 21](#_Toc17617732)

[**6.3 清除会话** 22](#_Toc17617733)

[**6.4 更新遗嘱数据的过程** 22](#_Toc17617734)

[**6.5 主题名称注册过程** 22](#_Toc17617735)

[**6.6 客户端的发布过程** 23](#_Toc17617736)

[**6.7 预定义的主题ID和短主题名称** 23](#_Toc17617737)

[**6.8 发布的QoS级别-1** 24](#_Toc17617738)

[**6.9 客户端主题的订阅/取消订阅过程** 24](#_Toc17617739)

[**6.10 网关的发布过程** 25](#_Toc17617740)

[**6.11 保持连接和PING过程** 25](#_Toc17617741)

[**6.12 客户端的断开过程** 25](#_Toc17617742)

[**6.13 客户端的重传过程** 26](#_Toc17617743)

[**6.14 对休眠客户端的支持** 26](#_Toc17617744)

[7 实施说明 28](#_Toc17617745)

[**7.1 支持QoS等级-1** 28](#_Toc17617746)

[**7.2 定时器和计数器的“最佳实践”值** 28](#_Toc17617747)

[**7.3 主题ID到主题名称的映射** 28](#_Toc17617748)

[**7.4 ZigBee相关问题** 29](#_Toc17617749)

**1. 更改和修订历史**

**1.1 版本1.0，2007年11月29日**

* 初始版本。

**1.2 版本1.1，2008年6月5日**

* 新功能：支持休眠设备。

**1.3 版本1.2，2011年5月20日**

* 新功能：支持大于255个字节的消息长度。
* 转发器封装的格式更改为Nicholas O'Leary提出的格式（nick [oleary@uk.ibm.com](mailto:oleary@uk.ibm.com)）。
* 添加了返回代码值“0x03已拒绝，不支持”。
* 字段“ReturnCode”添加到消息WILLTOPICRESP和WILLMSGRESP中。

**2. 简介**

近期无线传感器网络（WSN）的热度日益增加，无论是商业还是技术，原因是它简单，低成本和易于部署。 这些网络可以用于不同的目的， 从测量和检测，到自动化和过程控制。 典型的无线传感网络由大量的电池供电的传感器和执行器（SA）组成，这些器件通常只有有限的存储和处理能力。 由于SA节点的数量通常非常大，有线基础设施的部署成本昂贵，所以这些设备的无线通信很重要 。 这样的网络本质上是非常动态的：无线链路可能随时暂时中断，节点可能会失效并经常更换。 在这种情况下，与个别节点使用地址进行通信的传统方法可能成为一场噩梦。 应用程序驻留在固定网络上，与无线SA设备的交互，需要管理和维护与大量节点设备的通信。 在大多数情况下，他们不需要知道提供信息设备的地址或身份 ，但对数据的内容更感兴趣。 例如，资产跟踪应用，更感兴趣的是某个资产的当前位置，而不是提供该资产位置信息的GPS接收器的网络地址。 此外，一些应用可能对相同的传感器数据感兴趣，但出于不同的目的。在这种情况下，SA节点将需要并行的管理和维护具有多个应用程序的通信装置。 这可能会超过简单和低成本SA设备的有限功能。

另一个问题是网络之间的寻址方案的差异。 例如，基于TCP/IP的网络上的应用程序，如何寻址基于ZigBee无线网络的SA设备？

可以通过使用以消息为中心的通信方法来克服上述问题，其中信息发送给接收方，不是基于它们的网络地址，而是基于信息内容的功能和它们的兴趣。 一个广为所知的消息为中心系统是“发布、订阅”消息系统，已经在企业网络中广泛使用了，主要是由于它们的可扩展性和支持动态网络拓扑。 将企业发布/订阅系统扩展到无线传感网络，也可以实现无线传感网络无缝集成到企业网络中，从而使SA收集的数据，可提供给任何其他企业应用，并允许从任何企业应用中控制SA。 这可以通过使用MQTT协议来实现，该协议是开放的轻量级发布/订阅协议，专为机器对机器和移动应用场景而设计。 它针对带宽成本非常高的网络和通信不可靠的网络专门做了优化。 但是，MQTT需要基于底层网络，例如TCP/IP，提供有序的可靠连接，但这种网络对于非常简单、占用空间小、成本低的无线SA等设备来说还是太复杂了。

本文档的目的是详细描述MQTT-SN协议，一种用于无线传感器网络的发布/订阅协议。 MQTT-SN可以被视为MQTT的一个版本，它适应无线通信环境的特性。因为容易衰落和受到干扰，无线射频链路通常具有比有线链路更高的错误率。 它们的传输速率也较低。 例如， 基于IEEE 802.15.4标准的无线传感网络在2.4GHz频带中提供250kbit/s的最大带宽。 而且，为了抵抗传输错误，它们的分包长度非常短。 在IEEE的802.15.4定义中，物理层的数据包长度限制为128字节。 这128个字节中的一半还会被必要的网络传输信息占据， 比如MAC层，网络层，安全等。

MQTT-SN还针对具有有限处理能力和存储资源的低成本电池供电设备进行了优化 。

MQTT-SN最初开发用于在ZigBee APS层之上运行 。 ZigBee是开放工业联盟旨在为无线传感器网络定义一个开放的全球通信标准。为了全球化，ZigBee选择IEEE标准802.15.4作为PHY层和MAC层的协议，在此标准上增加了所需的网络，安全层和应用层，从而提供了不同厂商的产品之间的互操作性 。

MQTT-SN的设计使其无法与底层网络服务相关联。任何在某一节点和特定节点（网关）之间提供双向数据传输服务的网络能够支持MQTT-SN。 例如，一个允许源端点发送数据到特定目标端点的简单数据报服务就足够了。 广播数据传输服务仅在网关发现过程中需要。为了减少网关发现过程中的广播流量，期望MQTT-SN可以将所需的广播半径指示给底层网络。

**3. MQTT-SN与MQTT比较**

MQTT-SN设计为尽可能接近MQTT，但适应无线通信环境的特性，如低带宽，高链路故障，短消息长度等。它也为有限处理和存储资源的低成本，电池供电设备的实现进行了优化。

与MQTT相比，MQTT-SN具有以下不同之处：

1. CONNECT消息分为三条消息。 另外两个是可选的，用于将遗嘱主题和遗嘱消息传送到服务器。
2. 为了应对无线网络中的短消息长度和有限的传输带宽，PUBLISH消息中的主题名称由短的、两字节长的“主题ID”替换。 注册过程定义为允许客户端向服务器/网关注册主题名称并获取相应的主题ID。 它也用于相反的方向，以通知客户端主题名称和相对应的，将包含在接下来PUBLISH消息中的主题ID。
3. 引入了“预定义”主题ID和“短”主题名称，无需注册。 预定义的主题ID也是主题名称的两个字节的替换，它们映射到的主题名称，是客户端的应用程序和网关/服务器都预先知道的。 因此双方都可以使用预定义的主题ID，在“，不需要像正常”主题ID那样，上文提到的注册。短主题名称是刚刚好两字节的主体名称，它们足够短以至于可以和数据一起在PUBLISH消息中发送，和预定义主体id一样，短主题名称也不需要注册。
4. 发现过程可帮助没有预先配置的服务器/网关地址的客户端，发现运行着的服务器/网关的真实网络地址。 多个网关可以同时存在于单个无线网络中，可以工作在负载均衡模式或主从模式下协同工作。
5. “清除会话”的语义扩展到了遗嘱功能，即不仅仅是客户端的订阅是持久性，遗嘱主题和遗嘱消息也是。 客户端还可以在会话期间，修改遗嘱l主题和遗嘱消息。
6. 定义了一个新的离线保持活动过程，用来支持休眠的客户端。 通过这个程序， 电池供电的设备可以进入休眠状态，在该状态期间，所有发往它们的消息都被缓冲在服务器/网关处，并在他们醒来后发送给它们。

## 4. MQTT-SN架构

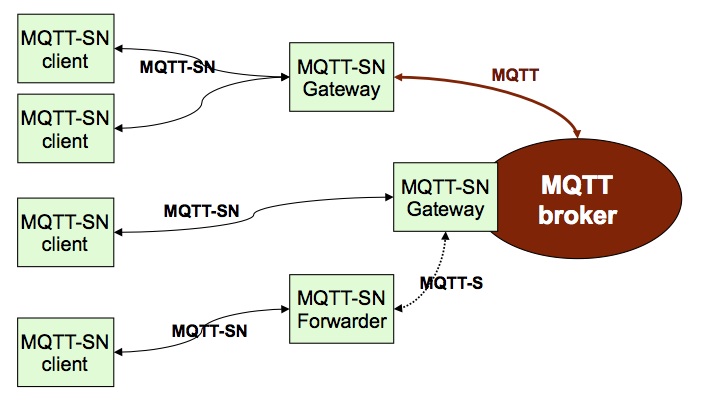


图1：MQTT-SN架构

MQTT-SN的体系结构如图1所示。有三种MQTT-SN组件，MQTT-SN 客户端，MQTT-SN网关（GW）和MQTT-SN转发器。 MQTT-SN客户端使用MQTT-SN协议，通过MQTT-SN网关，将自身连接到MQTT服务器。一个 MQTT-SN网关可以集成或不集成 MQTT服务器。 在独立网关的情况下，MQTT-SN网关和MQTT服务器之间使用MQTT协议。 其主要功能是MQTT和MQTT-SN协议之间的转换。

如果MQTT-SN客户端没有直接连接到网关所在的网络，它们也可以通过转发器访问网关。 转发器简单地封装它在无线侧接收的MQTT-SN帧，并将它们没有变化的发送到网关。 在相反的方向上，它也无变化的解封装从网关接收的帧，将它们发送给客户端。

根据网关如何在MQTT和MQTT-SN之间执行协议转换，我们可以区分 两种类型的网关，即透传网关和聚合网关，见图2。在下面部分解释说明。

**4.1 透传网关**

对于每个MQTT-SN客户端的连接，透传网关都将建立和维护与MQTT服务器的单独的连接。 这种MQTT连接专门用于端到端和几乎透明的客户端和服务器之间进行消息交换。 网关和服务器之间的连接数，与网关和MQTT-SN客户端之间的连接数一样多。 透传网关将执行两个协议之间的“语法”翻译。 由于MQTT-SN客户端和MQTT服务器之间的所有消息交换都是端到端的，服务器实现的所有功能和特性都可以向客户端提供。

虽然与聚合网关相比，透明网关的实现更简单，但是它要求MQTT服务器为每个活动客户端支持单独的连接。 一些MQTT服务器的实现可能会对并发连接数有限制。

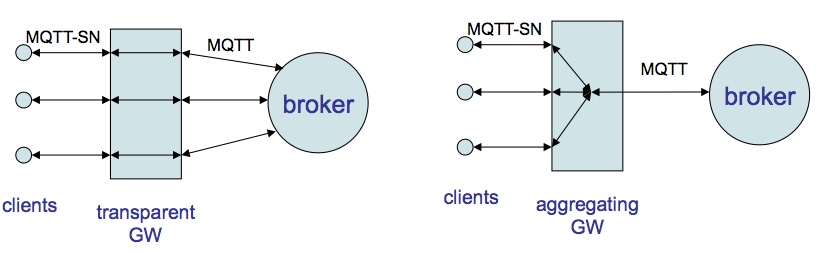


图2：透传网关和聚合网关

**4.2 聚合网关**

聚合网关只有一个连接到MQTT服务器，而不是为每个连接的客户端建立单独的MQTT连接。 MQTT-SN客户端和聚合网关之间的所有消息交换都终止在网关。 然后网关决定将哪些信息进一步提供给服务器。 虽然它的实现比透传网关更复杂，聚合网关在有大量的SA无线传感网络的情况下可能会更有用，因为它减少了服务器必须同时支持的MQTT连接数。

**5. 消息格式**

**5.1 基本消息格式**

|  |  |
| --- | --- |
| **消息头** | **消息可变部分** |
| 2或4字节 | n字节 |

表1：基本消息格式

MQTT-SN消息的基本格式如表1所示。MQTT-SN消息由两部分组成： 一个2或4字节长的头和一个可选的可变部分。 消息头始终存在并包含确定的字段，可变部分是否存在和可变部分内容，取决于消息的类型。

**5.2 消息头**

消息头的格式如表2所示。

|  |  |
| --- | --- |
| **长度** | **消息类型** |
| 1或3字节 | 1字节 |

表2：消息头

**5.2.1 长度**

“长度”字段为1或3字节，指明消息中包含的字节总数（包括长度字段本身）。

如果长度字段的第一个字节编码为“0x01”，则长度字段长度为3个字节。在这种情况下，接下来的两个字节指定消息的字节总数（高位在前）。 否则，长度字段只有1个字节，并指定包括自己在内的消息字节总数。

3字节格式允许编码最长65535字节的消息。 小于256个字节的消息可以使用1字节格式。

请注意，由于MQTT-SN不支持消息分段和重组，所以可以在网络中使用的最大消息长度，实际由该网络支持的最大数据包长度决定， 而不是由MQTT-SN编码的最大长度决定。

**5.2.2 消息类型**

“消息类型”字段长度为1个字节，并指定消息类型。它应设置为表3中的一个值。

|  |  |  |  |
| --- | --- | --- | --- |
| **字段值** | **消息类型** | **字段值** | **消息类型** |
| 0x00 | ADVERTISE | 0x01 | SEARCHGW |
| 0x02 | GWINFO | 0x03 | 保留 |
| 0x04 | CONNECT | 0x05 | CONNACK |
| 0x06 | WILLTOPICREQ | 0x07 | WILLTOPIC |
| 0x08 | WILLMSGREQ | 0x09 | WILLMSG |
| 0x0A | REGISTER | 0x0B | REGACK |
| 0x0C | PUBLISH | 0x0D | PUBACK |
| 0x0E | PUBCOMP | 0x0F | PUBREC |
| 0x10 | PUBREL | 0x11 | 保留 |
| 0x12 | SUBSCRIBE | 0x13 | SUBACK |
| 0x14 | UNSUBSCRIBE | 0x15 | UNSUBACK |
| 0x16 | PINGREQ | 0x17 | PINGRESP |
| 0x18 | DISCONNECT | 0x19 | 保留 |
| 0x1A | WILLTOPICUPD | 0x1B | WILLTOPICRESP |
| 0x1C | WILLMSGUPD | 0x1D | WILLMSGRESP |
| 0x1E-0xFD | 保留 | 0xFE | ??? |
| 0xFF | 保留 |  |  |

表3：基本类型字段值

**5.3 消息可变部分**

消息变量部分的内容取决于消息的类型。以下字段是消息定义在可变部分中的内容。

**5.3.1 ClientId**

与MQTT一样，ClientId字段具有可变长度，并且包含1-23个字符长，用来在服务端唯一的标识一个客户端。

**5.3.2 Data**

Data字段对应于MQTT PUBLISH消息的有效载荷。它有一个可变长度，包含要被发布的应用层数据。

**5.3.3 Duration**

“Duration”字段长度为2个字节，指定以秒为单位的持续时间。可编码的最大值大约18个小时。

**5.3.4 Flags**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DUP** | **QoS** | **Retain** | **Will** | **CleanSession** | **TopicIdType** |
| (bit 7) | (6,5) | (4) | (3) | (2) | (1,0) |

表4：Flags字段

“Flags”字段是1个字节，包含以下标志（参见表4）：

* DUP：与MQTT的含义相同，即如果第一次发送消息则设置为“0”; 如果设置为“1” 表示是重传的消息（仅在PUBLISH消息中使用）。
* QoS：消息质量，取值0( “0b00”), 1( “0b01”)和2( “0b10”)时，与MQTT含义相同。 值“0b11”表示一个新的QoS级别-1（仅用在由客户端发送的PUBLISH消息中）。
* Retain：与MQTT相同的含义（仅用在PUBLISH消息中）。
* Will：如果设置，则表示客户端正在询问Will主题和Will消息提示（仅用在 CONNECT消息中）。
* CleanSession：与MQTT的含义相同，但是对遗嘱主题和遗嘱消息进行了扩展（仅用在CONNECT消息中）。
* TopicIdType：表示此消息中的字段TopicId或TopicName是否包含法线 topic id（设置为“0b00”），预定义的topic id（设置为“0b01”）或短主题名称（设置为“0b10”）。 值“0b11”保留。 有关各种主题ID的定义，请参阅第3节和第6.7节。

**5.3.5 GwAdd**

"GwAdd"字段具有可变长度并包含网关的地址。它取决于运行MQTT-SN的物理网络操作并在该字段的第一个字节中表示。例如，在ZigBee网络中的网络地址长2个字节。

**5.3.6 GwId**

"GwId"字段长度为1个八位字节，唯一标识网关。

**5.3.7 MsgId**

"MsgId"字段长度为2个字节，对应于MQTT“Message ID”参数。它用于发送方匹配消息，以及相应的确认。

**5.3.8 ProtocolId**

"ProtocolId"长1个字节。它仅存在于CONNECT消息中，并且对应于MQTT协议名称和协议版本。

它编码为0x01。保留所有其他值。

**5.3.9 Radius**

"Radius"字段长度为1个字节，表示广播半径。值0x00表示“广播到网络中的所有节点“。

**5.3.10 ReturnCode**

表5中显示了1个字节长的"ReturnCode"字段的值和含义。

|  |  |
| --- | --- |
| **ReturnCode值** | **含义** |
| 0x00 | 接受 |
| 0x01 | 拒绝：拥塞 |
| 0x02 | 拒绝：无效的 topic ID |
| 0x03 | 拒绝：不支持 |
| 0x04-0xFF | 保留 |

表5：ReturnCode值

**5.3.11 TopicId**

“TopicId”字段长度为2个字节，包含topic ID的值。值“0x0000”和“0xFFFF”为保留，因此不应使用。

**5.3.12 TiopicName**

“TopicName”字段具有可变长度，包含指定了主题名称的UTF8编码字符串。

**5.3.13 WillMsg**

“WillMsg”字段具有可变长度，包含遗嘱消息。

**5.3.14 WillTopic**

“WillTopic”字段具有可变长度，包含遗嘱主题名称。

**5.4 消息格式**

本章节指定各个MQTT-SN消息的格式。所有消息默认消息长度字段是1字节。消息长度字段是3字节情况下的消息格式可以直接类比出来，因此没有提及。

**5.4.1 ADVERTISE**

|  |  |  |  |
| --- | --- | --- | --- |
| **Length** | **MsgType** | **GwId** | **Duration** |
| (byte 0) | (1) | (2) | (3,4) |

表6：ADVERTISE消息

“ADVERTISE”网关周期性地广播以通告其存在的消息。 发送下一个广播消息的时间间隔，在这个消息的Duration字段里说明。消息格式如表6所示：

* Length和MsgType：参见5.2节。
* GwId：发送此消息的网关的ID。
* Duration：此网关广播下一个ADVERTISE消息的时间间隔。

**5.4.2 SEARCHGW**

|  |  |  |
| --- | --- | --- |
| **Length** | **MsgType** | **Radius** |
| (byte 0) | (1) | (2) |

表7：SEARCHGW消息

SEARCHGW消息是客户端发出的广播消息，用来搜索网关。SEARCHGW的广播半径字段， 限制和取决于客户端部署的密度。例如，一个非常密集的网络，其中每个MQTT-SN客户端在1跳传输中彼此可达，广播半径只需要1跳。SEARCHGW消息的格式如表7所示：

* Length和MsgType：参见5.2节。
* Radius：消息的广播半径。

当MQTT-SN给出该消息时，广播半径也用来指示给底层网络层的传输。

**5.4.3 GWINFO**

|  |  |  |  |
| --- | --- | --- | --- |
| **Length** | **MsgType** | **GwId** | **GwAdd\*** |
| (byte 0) | (1) | (2) | (3,n) |

(\*)只有在消息由客户端发送时出现

表8：ADVERTISE消息

GWINFO消息作为对SEARCHGW消息的响应，在底层的广播服务中使用，广播半径如SEARCHGW消息中所示。

如果由网关发送，则仅包含发送GW的id， 否则，如果由客户端发送，它还包括网关的地址。如表8所示：

* Length和MsgType：参见5.2节。
* GwId：网关的id。
* GwAdd：网关的地址，可选，仅在客户端发送消息时包含。

与SEARCHGW消息一样，传输消息时，此消息的广播半径也指示给底层网络。

**5.4.4 CONNECT**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Length** | **MsgType** | **Flags** | **ProtocolId** | **Duration** | **ClientId** |
| (byte 0) | (1) | (2) | (3) | (4,5) | (6:n) |

表9：CONNECT消息

CONNECT消息由客户端发送以建立连接。格式如表9所示：

* Length和MsgType：参见5.2节。
* Flags：
  + DUP，QoS，Retain，TopicIdType：未使用。
  + Will：如果设置，则表示客户端正在请求遗嘱主题和遗嘱l消息。
  + CleanSession：与MQTT的含义相同，但是对遗嘱主题和遗嘱消息进行了扩展（请参阅 第6.3节）。
* ProtocolId：对应于MQTT CONNECT消息的“协议名称”和“协议版本”字段。
* Duration：与MQTT相同，Keep Alive计时器的值。
* ClientId：与MQTT相同，客户端ID，它是1-23个字符长的唯一字符串，为服务器标识一个唯一的客户端。

**5.4.5 CONNACK**

|  |  |  |
| --- | --- | --- |
| **Length** | **MsgType** | **ReturnCode** |
| (byte 0) | (1) | (2) |

表10：CONNACK消息

CONNACK消息由服务端发送，响应来自客户端的连接请求。格式如表10所示：

* Length和MsgType：参见5.2节。
* ReturnCode：根据表5编码。

**5.4.6 WILLTOPICREQ**

|  |  |
| --- | --- |
| **Length** | **MsgType** |
| (byte 0) | (1) |

表11：WILLTOPICREQ和WILLMSGREQ消息

WILLTOPICREQ消息由网关发送，请求客户端发送遗嘱主题名称。 它只有一个标题而没有变量部分，格式如表11所示。

**5.4.7 WILLTOPIC**

|  |  |  |  |
| --- | --- | --- | --- |
| **Length** | **MsgType** | **Flags** | **WillTopic** |
| (byte 0) | (1) | (2) | (3:n) |

表12：WILLTOPIC消息

WILLTOPIC消息由客户端发送，作为对WILLTOPICREQ消息的响应，传送它的遗嘱主题到网关。其格式如表12所示：

* Length和MsgType：参见5.2节。
* Flags：
  + DUP：不使用。
  + QoS：与MQTT相同，包含遗嘱 QoS
  + Retain：与MQTT相同，包含遗嘱 Retain标志
  + Will：不使用
  + CleanSession：不使用
  + TopicIdType：不使用。
* WillTopic：包含遗嘱主题名称。

空WILLTOPIC消息没有Flags和WillTopic字段（即它只有2个字节长）。 客户端使用它来删除存储在服务器中的遗嘱主题和遗嘱消息，请参阅6.4章节。

**5.4.8 WILLMSGREQ**

WILLMSGREQ消息由网关发送，请求客户端发送遗嘱消息。

它的格式是如表11所示：只有一个头部而没有可变部分。

**5.4.9 WILLMSG**

|  |  |  |
| --- | --- | --- |
| **Length** | **MsgType** | **WillMsg** |
| (byte 0) | (1) | (2:n) |

表13：WILLMSG消息

WILLMSG消息由客户端发送，作为对WILLMSGREQ的响应，用来传送遗嘱消息 到网关。其格式如表13所示：

* Length和MsgType：参见5.2节。
* WillMsg：包含遗嘱消息。

**5.4.10 REGISTER**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Length** | **MsgType** | **TopicId** | **MsgId** | **TopicName** |
| (byte 0) | (1) | (2,3) | (4,5) | (6:n) |

表14：REGISTER消息

REGISTER消息由客户端发送到网关，用于请求主题名称对应的主题id值。

它也可以由网关发送，以通知客户端它已分配给所包括的主题名称的主题id值。它的格式如表14所示：

* Length和MsgType：参见5.2节。
* TopicId：如果客户端发送，则编码为0x0000且不相干。如果由网关发送，它包含 分配给TopicName主题名称字段对应的主题ID。
* MsgId：应该被编码，便于在REGACK消息中对应。
* TopicName：主题名称。

**5.4.11 REGACK**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Length** | **MsgType** | **TopicId** | **MsgId** | **ReturnCode** |
| (byte 0) | (1) | (2,3) | (4,5) | (6) |

表15：REGACK消息

REGACK消息由客户端或网关发送，作为对一个REGISTER的接收和处理确认。其格式如表15所示：

* Length和MsgType：参见5.2节。
* TopicId：在PUBLISH消息中用作主题ID的值。
* MsgId：与相应的REGISTER消息中包含的值相同。
* ReturnCode：“已接受”或拒绝原因。

**5.4.12 PUBLISH**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Length** | **MsgType** | **Flags** | **TopicId** | **MsgId** | **Data** |
| (byte 0) | (1) | (2) | (3,4) | (5,6) | (7:n) |

表16：PUBLISH消息

客户端和网关都使用此消息来发布特定主题的数据。其格式如表16中所示：

* Length和MsgType：参见5.2节。
* Flags：
  + DUP：与MQTT相同，表示是否第一次发送消息。
  + QoS：与MQTT相同，包含此PUBLISH消息的QoS级别。
  + Retain：与MQTT相同，包含保留消息标志。
  + Will：不使用。
  + CleanSession：不使用。
  + TopicIdType：表示TopicId字段中包含的主题ID的类型。
* TopicId：包含发布数据的主题id值或短主题名称。
* MsgId：与MQTT“消息ID”的含义相同; 仅在QoS级别1和2的情况下使用，否则 编码为0x0000。
* 数据：要发布的数据。

**5.4.13 PUBACK**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Length** | **MsgType** | **TopicId** | **MsgId** | **ReturnCode** |
| (byte 0) | (1) | (2,3) | (4,5) | (6) |

表17：PUBACK消息

PUBACK消息由网关或客户端发送，作为对发布消息的接收和处理的确认 （在QoS级别1或2的情况下）。

它也可以作为对PUBLISH消息发送 出错的响应情况，然后在ReturnCode字段中指示错误原因。其格式如表17所示：

* Length和MsgType：参见5.2节。
* TopicId：与相应PUBLISH消息中包含的值相同的值。
* MsgId：与相应PUBLISH消息中包含的值相同的值。
* ReturnCode：“已接受”或拒绝原因。

**5.4.14 PUBREC，PUBREL和PUBCOMP**

|  |  |  |
| --- | --- | --- |
| **Length** | **MsgType** | **MsgId** |
| (byte 0) | (1) | (2,3) |

表18：PUBREC，PUBREL和PUBCOM消息

与MQTT一样，PUBREC，PUBREL和PUBCOMP消息与PUBLISH一起使用（QoS级别2的消息中）。 它们的格式如表18所示：

* Length和MsgType：参见5.2节。
* MsgId：与相应PUBLISH消息中包含的值相同。

**5.4.15 SUBSCRIBE**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Length** | **MsgType** | **Flags** | **MsgId** | **TopicName 或 TopicId** |
| (byte 0) | (1) | (2) | (3,4) | (5:n) 或 (5,6) |

表19：SUBSCRIBE消息

SUBSCRIBE消息由客户端发送，来订阅某个主题名称。其格式如表19所示：

* Length和MsgType：参见5.2节。
* Flags：
  + DUP：与MQTT相同，表示是否第一次发送消息。
  + QoS：与MQTT相同，包含此主题的请求QoS级别。
  + Retain：不使用。
  + Will：不使用。
  + CleanSession：不使用
  + TopicIdType：表示主题的类型，即“0b00” 主题名称，“0b01”预定义主题ID，“0b10”短主题名称和“0b11”保留。
* MsgId：应被编码，以便于相应的SUBACK消息进行对应。
* TopicName或TopicId：包含TopicIdType中指示的主题名称，主题ID或短主题名称。

**5.4.16 SUBACK**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Length** | **MsgType** | **Flags** | **TopicId** | **MsgId** | **ReturnCode** |
| (byte 0) | (1) | (2) | (3,4) | (5,6) | (7) |

表20：SUBACK消息

SUBACK消息由网关发送给客户端，作为对SUBSCRIBE消息接收和处理的确认。其格式如表20所示：

* Length和MsgType：参见5.2节。
* Flags：
  + DUP：不使用。
  + QoS：与MQTT相同，包含授予的QoS级别。
  + Retain：不使用。
  + Will：不使用。
  + CleanSession：不使用。
  + TopicIdType：不使用。
* TopicId：在“已接受”的情况下，网关在发送PUBLISH时将用此TopicId的值 向客户端发送消息（在订阅短主题名称或主题名称时忽略）。
* MsgId：与相应订阅消息中包含的值相同的值。
* ReturnCode：“已接受”或拒绝原因。

**5.4.17 UNSUBSCRIBE**

UNSUBSCRIBE消息由客户端向网关发送，以取消订阅主题。它的格式是如表19所示：

* Length和MsgType：参见5.2节。
* Flags：
  + DUP：与MQTT相同，表示是否第一次发送消息。
  + QoS：与MQTT相同，包含此主题的请求QoS级别。
  + Retain：不使用。
  + Will：不使用。
  + CleanSession：不使用
  + TopicIdType：表示主题的类型，即“0b00” 主题名称，“0b01”预定义主题ID，“0b10”短主题名称和“0b11”保留。
* MsgId：应被编码，以便于相应的SUBACK消息进行对应。
* TopicName或TopicId：包含TopicIdType中指示的主题名称，主题ID或短主题名称。

**5.4.18 UNSUBACK**

|  |  |  |
| --- | --- | --- |
| **Length** | **MsgType** | **MsgId** |
| (byte 0) | (1) | (2,3) |

表21：UNSUBACK消息

UNSUBACK消息由网关发送，以确认接收和处理取消订阅消息。其格式如表21所示：

* Length和MsgType：参见5.2节。
* MsgId：与相应的UNSUBSCRIBE消息中包含相同的值。

**5.4.19 PINGREQ**

|  |  |  |
| --- | --- | --- |
| **Length** | **MsgType** | **ClientId(可选)** |
| (byte 0) | (1) | (2:n) |

表22：PINGREQ消息

与MQTT一样，PINGREQ代表一个“你还活着吗?”消息，由一个 连接的客户端发送或接收。其格式如表22所示：

* Length和MsgType：参见5.2节。
* ClientId：客户端ID，这个字段是可选的，出现在客户端由“休眠”状态切换 到“唤醒”状态并在等待服务器/网关发送消息时。 细节说明请参见第6.14节。

**5.4.20 PINGRESP**

|  |  |
| --- | --- |
| **Length** | **MsgType** |
| (byte 0) | (1) |

表23：PINGRESP消息

与MQTT一样，PINGRESP消息是对PINGREQ消息的响应，意思是“是的，我还活着”。

保活消息以任一方向流动，连接的客户端或网关都可以发送。

它的格式是如表23所示：它只有一个头部而没有可变部分。

此外，网关发送PINGRESP消息以通知睡眠客户端，它没有该客户端更多的缓存消息了，有关更多详细信息，请参见第6.14节。

**5.4.21 DISCONNECT**

|  |  |  |
| --- | --- | --- |
| **Length** | **MsgType** | **Duration(可选)** |
| (byte 0) | (1) | (2:3) |

表24：DISCONNECT消息

DISCONNECT消息的格式如表24所示：

* Length和MsgType：参见5.2节。
* Duration：睡眠定时器的值。 此字段是可选的，出现在客户端由“休眠”状态 想要进入“睡眠”状态时，请参阅第6.14节了解更多详情。

与MQTT一样，客户端发送DISCONNECT表示它要关闭连接。

网关将通过向客户端返回一个DISCONNECT消息，来确认收到该消息。

服务器或网关也可以向客户端发送DISCONNECT消息，例如，在网关由于错误而无法映射一个收到的消息到客户端。客户端收到此类DISCONNECT消息后，应尝试通过向网关或服务器发送CONNECT消息再次建立连接。所有这些情况下，DISCONNECT消息都不包含Duration字段。

当客户端想要进入“睡眠”状态时，会发送带有Duration字段的DISCONNECT消息。 网关也通过发送DISCONNECT消息确认收到此消息（没有Duration字段）。

**5.4.22 WILLTOPICUPD**

|  |  |  |  |
| --- | --- | --- | --- |
| **Length** | **MsgType** | **Flags** | **WillTopic** |
| (byte 0) | (1) | (2) | (3:n) |

表25：WILLTOPICUPD消息

WILLTOPICUPD消息由客户端发送，用来更新其存储在网关或服务器中的遗嘱主题名称。它的格式如表25所示：

* Length和MsgType：参见5.2节。
* Flags：
  + DUP：不使用。
  + QoS：与MQTT相同，遗嘱消息质量等级。
  + Retain：与MQTT相同，保留标志。
  + Will：不使用。
  + CleanSession：不使用。
  + TopicIdType：不使用。
* WillTopic：遗嘱主题名称。

空WILLTOPICUPD消息是没有Flags和WillTopic字段（即它恰好是2个字节长）。客户端使用它来删除存储在网关或服务器中的遗嘱主题和遗嘱消息。

**5.4.23 WILLMSGUPD**

|  |  |  |
| --- | --- | --- |
| **Length** | **MsgType** | **WillTopic** |
| (byte 0) | (1) | (2:n) |

表26：WILLMSGUPD消息

WILLMSGUPD消息由客户端发送，用来更新其存储在网关或服务器中的遗嘱消息。它的格式如表26所示：

* Length和MsgType：参见5.2节。
* WillMsg：遗嘱消息。

**5.4.24 WILLTOPICRESP**

|  |  |  |
| --- | --- | --- |
| **Length** | **MsgType** | **ReturnCod** |
| (byte 0) | (1) | (2) |

表27：WILLTOPICRESP和WILLMSGRESP消息

WILLTOPICRESP消息由网关发送，用来对WILLTOPICUPD消息的接收和处理确认。其格式如表27所示：

* Length和MsgType：参见5.2节。
* ReturnCode：“已接受”或拒绝原因。

**5.4.25 WILLMSGRESP**

WILLMSGRESP消息由网关发送，用来对WILLMSGUPD消息的接收和处理确认。其格式如表27所示：

* Length和MsgType：参见5.2节。
* ReturnCode：“已接受”或拒绝原因。

**5.5 转发器封装**

如第4节所述，当网关不是直接连接在无线传感器网络中时，MQTT-SN客户端也可以通过转发器访问网关。

转发器简单地封装它在无线端接收的MQTT-SN消息帧，并将它们不变地转发给网关。

在相反的方向上，转发器解封装它从网关接收的消息帧，并也保持不变的将它们发送给客户端。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Length** | **MsgType** | **Ctrl** | **Wireless Node Id** | **MQTT-SN message** |
| (byte 0) | (1) | (2) | (3:n) | (n+1:m) |

表28：封装的MQTT-SN帧格式

封装的MQTT-SN帧的格式如表28所示：

* Length：1个字节长，指定直到“Wireless Node Id”字段末尾的字节数（包括 长度字段本身）
* MsgType：编码为“0xFE”，参见表3。
* Ctrl：包含网关和转发器之间交换的控制信息。它的格式如表29所示：
  + Radius：广播半径（仅在网关到转发器的方向上有效）。
  + 保留所有剩余比特。
* Wireless Node Id：已发送或需要接收封装的MQTT-SN消息的无线节点的标识。此标识与无线节点的实际地址之间的映射关系由转发器保存(如果需要的话)。
* MQTT-SN Message：根据表1编码的MQTT-SN消息。

|  |  |
| --- | --- |
| **保留** | **广播半径** |
| (bit 7:2) | (bit 1,0) |

表29：Ctrl字节格式

## 6. 功能描述

MQTT-SN的一个重要设计点是尽可能接近MQTT。因此，所有协议语义 应尽可能保持与MQTT定义的相同。

在下文中，我们将重点关注那些相对于MQTT新增或修改的点。

**6.1 网关的广播和发现**

此过程是全新的，在MQTT中不存在。

网关可以通过周期性地向所有连到网络中的设备，广播ADVERTISE消息来宣告其存在。网关应该仅通告它自己的存在（如果网关连接到服务器，或者它本身就是一个服务器）。

多个网关可以在同一网络中同时处于活动状态。在这种情况下，他们会有不同的id，由客户端决定它想要连接哪个网关。但是在任何时候客户端都只允许连接到一个网关。

客户端应维护活动的网关列表及其网络地址。这个列表通过收到的ADVERTISE消息和GWINFO消息填充和更新。

网关发送下一个ADVERTISE消息的时间间隔TADV，由ADVERTISE消息的Duration字段指示。客户端可以使用此信息来监视网关的可用性。 例如，如果它没有接收到来自网关的ADVERTISE消息连续NADV次数，它可以假设网关已经关闭并将其从活动网关列表中删除。同样，处于备用状态的网关将变为活动状态（即开始发送ADVERTISE消息）如果它连续几次错过 来自某个网关的ADVERTISE消息，替代不可用的网关。

由于ADVERTISE消息被广播到整个无线网络中，因此网关发送的两个ADVERTISE消息时间间隔TADV应该足够大（例如大于15分钟）以避免网络中的带宽拥塞。

TADV取值过大，将导致寻找网关的新客户端等待很长时间。为了缩短该等待时间，客户端可以广播SEARCHGW消息。为了防止几个客户端几乎同时开始搜索网关时产生的广播风暴，发送SEARCHGW消息应 延迟0到TSEARCHGW之间的随机时间。客户端如果在此延迟时间内收到另一个客户端发送的SEARCHGW消息，它将取消SEARCHGW的发送，并且表现得好像这个SEARCHGW消息是自己发送的。

SEARCHGW消息的广播半径Rb是有限的，例如在密集部署情况下的单跳MQTT-SN客户端。

收到SEARCHGW消息后，网关将回复包含自己id的GWINFO消息。同样的，如果客户端在其活动网关列表中至少有一个活动网关，它也会使用GWINFO消息进行应答。如果客户端在其列表中有多个网关，它则从其中选择一个网关并将该信息放在GWINFO消息应答。

与SEARCHGW消息一样，GWINFO消息以相同的半径Rb广播，这个值 在SEARCHGW消息中获取。 当传输这两种消息时，广播半径Rb也被传递到底层传播。

为了优先考虑网关，客户端将延迟一段随机时间TGWINFO发送GWINFO消息。如果在此延迟时间内客户端收到GWINFO消息，它将取消发送GWINFO消息。

在没有响应的情况下，可以重传SEARCHGW消息。在这种情况下，两个连续的SEARCHGW消息的时间间隔应该以指数方式增加。

**6.2 客户端的连接设置**

与MQTT一样，MQTT-SN客户端需要先与网关建立连接，然后才能与那个网关交换信息。与网关的建立连接的过程在图3中表示，其中假设客户端请求网关传输遗嘱主题和遗嘱消息。这种请求 通过设置CONNECT消息的Will标志来表示。然后客户端在收到网关相应的请求消息WILLTOPICREQ和WILLMSGREQ时，将这两条信息发送给网关。流程终止于网关发送的CONNACK消息。

如果Will标志未设置，则网关直接用CONNACK消息应答。

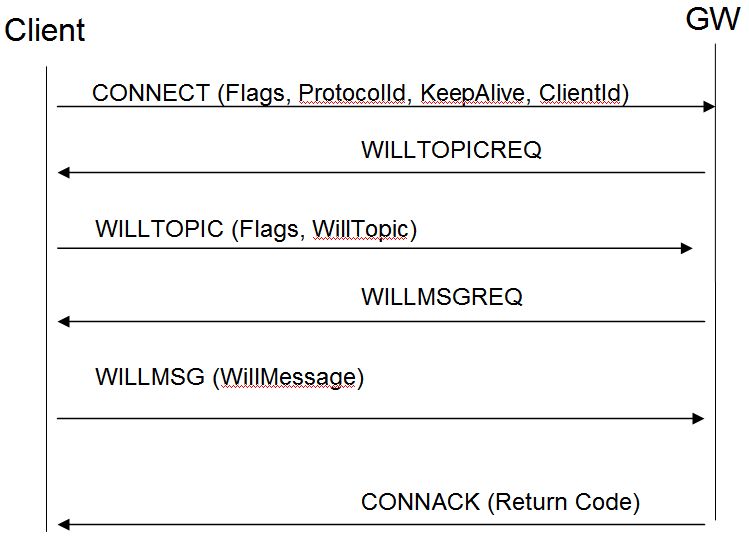


图3：连接过程

如果网关无法接受连接请求（例如由于拥塞或不支持客户端在CONNECT消息中指示的特性），网关返回具有拒绝原因的CONNACK消息。

**6.3 清除会话**

使用MQTT，当客户端断开连接时，不会删除其订阅。它们对于新的有效连接是持久的，直到客户端明确取消订阅，或客户端建立新连接设置“clean session”标志。

在MQTT-SN中，“清除会话”的含义扩展到遗嘱功能，即不仅是订阅是持久性的，遗嘱主题和遗嘱消息也是。CONNECT消息中的两个标志“CleanSession”和“Will”具有以下含义：

* CleanSession = true，Will = true：网关将删除所有与客户端相关的订阅和遗嘱数据， 并开始提示新的遗嘱主题和遗嘱消息。
* CleanSession = true，Will = false：网关将删除所有与客户端相关的订阅和遗嘱数据，并返回CONNACK（不提示遗嘱主题和遗嘱消息）。
* CleanSession = false，Will = true：网关保留所有存储的客户端数据，但提示输入新的遗嘱主题和遗嘱消息。新接收的遗嘱数据将覆盖存储的遗嘱数据。
* CleanSession = false，Will = false：网关保留所有存储的客户端数据并返回CONNACK（不提示遗嘱主题和遗嘱消息）。

请注意，如果客户端想要在连接设置时仅删除遗嘱数据，则可以发送CONNECT消息时使用“CleanSession = false”和“Will = true”，并在收到提示时向网关发送空的WILLTOPIC消息来实现。它还可以发送带有“CleanSession = false”和“Will = false”的CONNECT消息，并使用第6.4节中删除或修改遗嘱数据的过程。

**6.4 更新遗嘱数据的过程**

在连接期间的任何时候，客户端都可以通过发送WILLTOPICUPD或者WILLMSGUPD消息，来更新存储在网关中的遗嘱数据。 这两条消息中包含的信息将覆盖存储在网关中的相应数据。两个消息都由网关确认。两条消息都可以 彼此独立使用。

请注意，空的WILLTOPICUPD消息将删除存储在网关中的遗嘱主题和遗嘱消息。

**6.5 主题名称注册过程**

由于无线传感器网络中的带宽有限且消息有效载荷较小，因此不会像MQTT中那样，数据和主题名在一条发布消息中一起使用。为此引入了注册程序，允许客户端和网关可以在发布消息之前，通知对方短主题ID和主题名称的对应关系，然后使用短主题ID发布消息。

为了注册主题名称，客户端向网关发送REGISTER消息。如果注册可以接受， 网关将主题Id分配给接收到的主题名称，并将REGACK消息返回给客户端。 如果注册无法接受，REGACK也会发送给客户端，并在“ReturnCode”字段中编码失败的原因。

在收到带有ReturnCode =“accepted”的REGACK消息后，客户端将使用分配的主题Id发布相应主题名称的数据。但是，如果REGACK包含拒绝代码，客户可以尝试再次注册。如果返回代码为“拒绝：拥塞”，则客户端应等待时间TWAIT后，再重新启动注册过程。

在任何时间点，客户端只能有一个未完成的REGISTER消息，即它必须等待一个REGACK消息后，才可以注册另一个主题名称。

如果网关想要通知客户端，有关主题名称和分配的主题ID的对应关系，则网关会向客户端发送REGISTER消息。便于网关稍后向客户端发布相应主题名称的消息。 这种情况会发生在，例如，当客户端重新连接而未设置“CleanSession”标志时，或者客户端订阅的主题包含＃或+等通配符时。

**6.6 客户端的发布过程**

在成功的注册一个主题名称到网关之后，客户端可以通过向网关发送PUBLISH消息，开始发布与这个注册主题相关的数据，PUBLISH消息中包含分配的主题ID。

MQTT中定义的所有三个的QoS级别及其相应的消息流，在MQTT-SN中都支持。唯一的差异是在PUBLISH消息中使用主题ID而不是主题名称。

无论请求的QoS级别如何，客户端都可以接收PUBACK消息，来查看对PUBLISH的接受情况，其中包含

* ReturnCode =“拒绝：无效主题ID”：在这种情况下，客户端需要再次注册主题名称，来发布与该主题名称相关的数据。
* ReturnCode =“拒绝：拥塞”：客户端应停止向网关发布至少TWAIT时间。

在任何时间点，客户端可能只有一个未完成的QoS级别1或2的PUBLISH消息，即它必须等待此PUBLISH消息交换的终止，然后才能启动新的QoS 1或QoS 2事务。

**6.7 预定义的主题ID和短主题名称**

如6.5节所述，主题ID是对基于字符串的主题名称的两字节的长替换。一个客户端需要使用REGISTER过程来通知网关它想要使用的主题名称，并从网关获取到对应的主题的ID。然后在发送到网关的PUBLISH消息中使用此主题ID。在相反的方向上，网关PUBLISH消息也包含一个2字节的主题ID（而不是基于字符串的主题名称），客户端通过以下方式获取到主题ID和主题名称的对应关系，要么是前订阅过程，要么是网关启动的注册过程。

“预定义”主题ID是这样的主题ID，它的主题名称映射，是客户端应用程序和网关都预先知道的。这在消息的Flags字段中指示。使用预定义主题ID时， 双方可以立即开始发送PUBLISH消息，不需要注册过程，就像“正常”主题ID一样。收到带有预定义主题ID的PUBLISH消息时， 如果其中主题名称的映射未知，接收方应返回带有ReturnCode = “拒绝：无效主题ID”的PUBACK消息。请注意，和正常主题ID的情况不同，此错误无法通过重新注册来解决。

如果客户端想要接收某个预定义的主题ID的PUBLISH消息，它仍然需要订阅该主题ID。为了避免一个预定义主题ID和一个两字节的短主题名称之间产生混淆，订阅消息包含一个标志，指示它是订阅短主题名称还是预定义的主题ID。

“短”主题名称是具有固定两个字节长度的主题名称。它可以和数据一起携带在PUBLISH消息中，因此短主题名称不需要注册过程。否则，全部适用于普通主题名称的规则也适用于短主题名称。但请注意，在订阅短主题名称时使用通配符是没有意义的 ，因为只有两个字符，无法定义有意义的名称层次结构。

**6.8 发布的QoS级别-1**

此功能是为非常简单的、不支持除此任何其他功能的客户端定义的。 没有建立连接也没有断开，没有注册或订阅。客户端只是向网关（网关地址由客户端事先知道）发送它的PUBLISH消息并忘记它们。它并不关心网关地址是否正确，网关是否存活，或者消息是否到达网关。

对于服务质量级别为-1的PUBLISH消息，仅允许以下参数值：

* QoS的标志：设置为“0b11”。
* TopicIdType标志：预定义主题ID为“0B01”，短主题名称为“0b10”。
* TopicId字段：预定义主题ID或短主题名称的值。
* Data字段：要发布的数据。

**6.9 客户端主题的订阅/取消订阅过程**

要订阅主题名称，客户端会向网关发送一条包含要订阅主题的SUBSCRIBE消息。如果网关接受订阅，则会为接收的主题名称分配主题ID，并将主题ID方在SUBACK消息中发给客户端。如果订阅无法接受，也发送SUBACK消息到客户端，拒绝原因在ReturnCode字段中编码。如果拒绝原因是“被拒绝：拥塞”，客户端应该等待TWAIT时间后，重新发送SUBSCRIBE消息到网关。

如果客户端订阅包含通配符的主题名称，返回的SUBACK消息将包含主题id值0x0000。 当要发送匹配主题名称的第一个PUBLISH消息时， 客户端网关使用注册过程，将使用的主题ID值通知客户端，另见第6.10节。

与客户端的PUBLISH过程类似，也可以为某些主题名称预定义主题ID。 短主题名称 也可以使用。在这两种情况下，客户端仍需要订阅这些预定义的主题ID或短主题名称。

要取消订阅，客户端会向网关发送UNSUBSCRIBE消息，然后由网关应 UNSUBACK消息。

对于注册过程，客户端同一时刻只能有一个订阅或一个UNSUBCRIBE事务。

**6.10 网关的发布过程**

与6.6节中描述的客户端发布过程类似，网关发送的PUBLISH消息中，包含之前在SUBACK消息中返回给客户端的主题ID值。

在发送PUBLISH消息之前，网关可以发送REGISTER消息，以通知客户端主题名称对应的主题ID值。这种情况发生在，例如，当客户端重新连接时没有清除会话时，或者使用通配符订阅主题名称时。收到REGISTER消息后，客户端回复REGACK消息。网关将等待收到REGACK消息后，再在向客户端发送PUBLISH消息。

客户端可以发送REGACK消息拒绝REGISTER消息，在消息中指示拒绝原因。这也表示客户端取消订阅了REGISTER消息中指示的主题名称。请注意，取消订阅带通配符的主题名称只能使用第6.9节中描述的取消订阅过程来完成，而不是拒绝REGISTER消息，因为REGISTER消息永远不包含带有通配符主题名称 。

如果客户端收到具有未知主题ID值的PUBLISH消息，则它将以带有ReturnCode =“已拒绝：无效主题ID”的PUBACK消息响应。这将触发网关删除或更正错误的主题ID分配。

请注意，如果主题名称或数据太长而无法放入REGISTER或PUBLISH消息中，则网关以静默方式中止发布过程，即不向受影响的订阅者发送警告。

**6.11 保持连接和PING过程**

与MQTT一样，Keep Alive计时器的值在CONNECT消息中指示。 客户应该在每个Keep Alive时间段内发送PINGREQ消息，网关通过PINGRESP确认信息。

类似地，客户端在收到来自它所连接的网关的PINGREQ消息时，应使用PINGRESP消息进行应答。否则，就意味着收到的PINGREQ消息被忽略了。

客户端应使用此过程来监督它所连接的网关的活跃性。 如果即使在多次重传PINGREQ之后，客户端也没有从网关接收PINGRESP消息，它应该首先尝试连接到另一个网关，然后再尝试重新连接到此网关（另请参阅 第6.13节）。请注意，因为客户端的保持活动计时器彼此不同步，如果网关故障，几乎没有同时发送的CONNECT消息风暴的危险，所有受影响的客户端都会转向新的网关。

**6.12 客户端的断开过程**

客户端向网关发送DISCONNECT消息以指示它将要关闭连接。在这之后，客户端如果想要重新与网关交换信息，它必须与网关建立新的连接。与MQTT类似，如果设置了CleanSession标志，发送DISCONNECT消息不会影响现有订阅和遗嘱消息。除非客户端明确的取消订阅，或者客户端删除或修改，或者客户端用CleanSession标志建立了新连接，否则它们是持久的。网关通过向客户端返回DISCONNECT消息来确认收到断开消息。

客户端也可能在未请求的情况下，收到网关发送的DISCONNECT消息。这种情况发生在，例如，当网关由于错误，而无法识别接收到的消息所属的客户端时。客户端收到这样的DISCONNECT消息后，应通过发送CONNECT消息，尝试再次与网关建立连接。

**6.13 客户端的重传过程**

所有向网关“单播”的消息（即使用网关的单播地址发送而不是广播），对消息预期的网关回复，由重试计时器Ťretry和重试计数器Nretry来监督。该重试计时器由客户端的在发送消息时启动，在收到网关预期的回复时。如果计时器超时还未收到预期的网关回复，则客户端重新发送该消息。在Nretry次重传之后，客户端中止该过程，并假定它与网关的MQTT-SN连接已断开。然后它尝试一次重新连接到当前网关，连接失败的话，它应该尝试连接到另一个网关。

**6.14 对休眠客户端的支持**

休眠客户端是工作在尽可能节省电源（电池供电）设备上的客户端。 这些设备只要处于非活动状态时，都需要进入休眠模式，并且在有数据发送或接收时唤醒 。服务器或者网关需要知道这些客户端的休眠状态，并将缓冲发送给他们的消息，以便以后在他们醒来时发送。

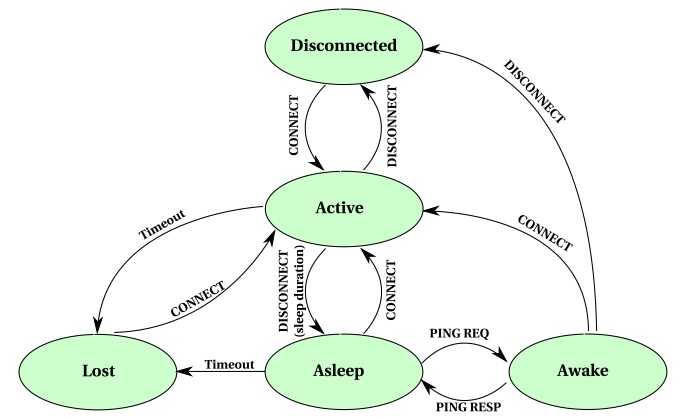


图4：客户端状态转换

如图4所示，从服务器或网关的角度来看，客户端可以是以下状态之一：活动，休眠，清醒，连接断开或丢失。当服务器或网关接收客户端发送的CONNECT消息时，客户端处于活动状态，如6.2节所述。该状态由服务器或网关使用“保持活动”计时器监督，如第6.11节所述。如果服务器或网关没有收到客户端的任何消息，持续时间超过保持活动持续时间（在CONNECT消息中指示）时，网关将客户端视为丢失并激活该客户端的遗嘱功能。

当服务器或网关收到没有duration字段的DISCONNECT消息时，客户端进入断开连接状态。 该状态不受服务器或网关的时间监督。

如果客户端想要休眠，它会发送包含休眠持续时间的DISCONNECT消息。服务器或网关 发送DISCONNECT消息来对客户端确认，并认为客户端处于休眠状态， 另请参见图5。休眠状态由服务器或网关根据指示的休眠时间监控。 如果超过休眠时间后，服务器或网关没有收到来自客户端的任何消息，服务器或网关将认为该客户端丢失，并且与保活过程一样，激活遗嘱特性。在休眠状态期间，需要发送到客户端的所有消息都缓存在服务器或网关。

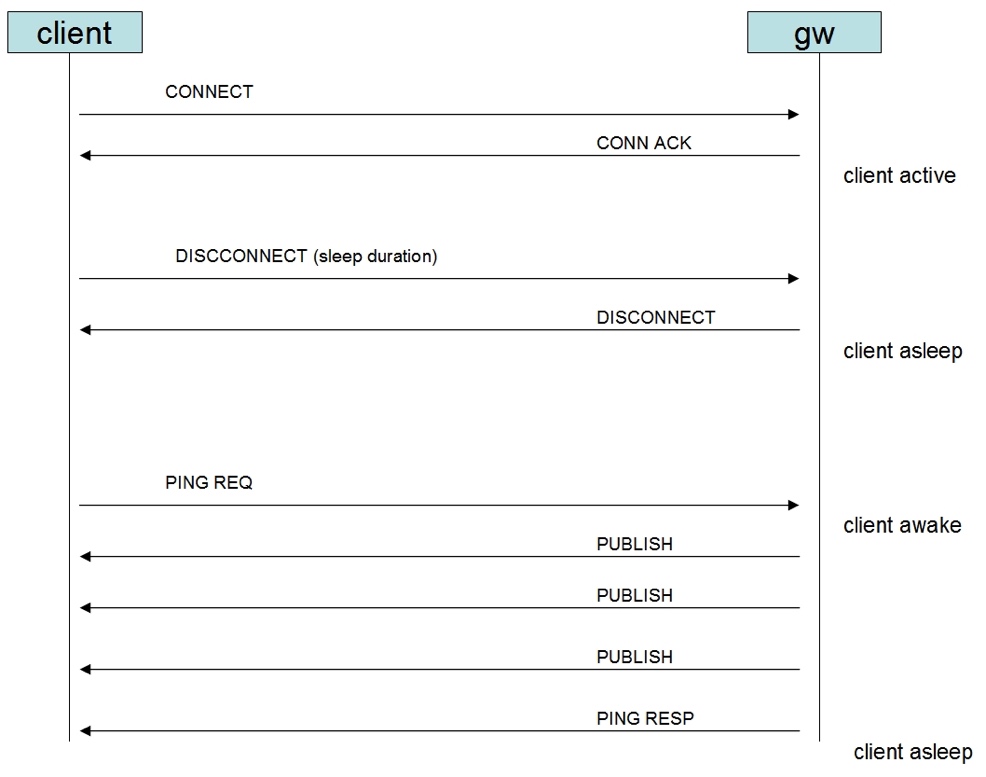


图5：休眠过程

当服务器或网关从客户端接收到PINGREQ消息时，停止休眠计时器。像CONNECT消息一样，此PINGREQ消息包含客户端ID。然后，所识别的客户端处于清醒状态。 如果服务器或网关缓存了该客户端的消息，它会将这些消息发送给客户端。向客户端的消息传输过程，由服务器或网关通过PINGRESP消息停止，也就是说，服务器或网关再次将客户端视为睡眠状态，并在发送PINGRESP消息后重新启动睡眠定时器。

如果服务器或网关没有为客户端缓冲的任何消息，则立即使用一个 PINGRESP消息进行应答，将客户端返回到睡眠状态，并重新启动该客户端的睡眠定时器。

在发送PINGREQ消息到服务器或网关之后，客户端使用6.13章节的“重传过程” 监督服务器或网关发送的消息的到达，即，客户端每收到一个PINGRESP以外的消息时，重启重试计时器，并在收到PINGRESP时停止计时器。如果计时器超时，客户端重传PINGREQ消息并重启定时器。为避免因过度重传PINGREQ消息导致的电池消耗（例如，如果它丢失了网关），客户端应有限的重传PINGREQ消息（例如通过重试计数器）并在达到限制但仍然没有收到PINGRESP消息时，返回休眠状态 。

在睡眠或唤醒状态，客户端可以通过发送CONNECT消息返回到活动状态，或通过发送正常的DISCONNECT消息（即没有duration字段）到断开状态。客户端还可以通过发送具有duration的新值的DISCONNECT消息来修改其睡眠持续时间。

请注意，只有在想要检查服务器或网关是否有缓冲的消息时，睡眠客户端才应进入唤醒状态 ，并尽快返回到睡眠状态而不发送任何消息到服务器或网关。否则，它应该通过向服务器或网关发送CONNECT消息返回到活动状态。

## 7 实施说明

**7.1 支持QoS等级-1**

因为客户端可以随时发送QoS等级为-1的PUBLISH消息（即使没有建立连接），一个转发网关需要为这些消息维护与服务器的专用MQTT连接。 一个聚合或混合网关可以使用任何聚合MQTT连接将这些消息转发到服务器。

**7.2 定时器和计数器的“最佳实践”值**

表30显示了本规范中定义的定时器和计数器的“最佳实践”值。

|  |  |
| --- | --- |
| **时间或次数** | **推荐值** |
| TADV | 大于15分钟 |
| NADV | 2-3 |
| TSEARCHGW | 5 秒 |
| TGWINFO | 5 秒 |
| TWAIT | 大于5分钟 |
| Tretry | 10-15秒 |
| Nretry | 3-5 |

表30：时间或次数的最佳实践值

服务器或网关上的睡眠和保持活动计时器的“容差”取决于客户端指示的持续时间。例如，对于大于1分钟的持续时间，计时器值应比指示值高10％ ，小于1分钟的持续时间应增加50％。

**7.3 主题ID到主题名称的映射**

强烈建议在网关中实现主题ID和主题名称之间的映射表，是每个客户端单独实现的（而不是所有客户端共享一个映射表），以减少来自某个客户端的错误的主题ID匹配了另一个客户端的有效主题ID的风险，从而导致发布错误的主题，这可能会带来灾难性的后果。

**7.4 ZigBee相关问题**

* 在ZigBee网络中，网关不需要由协调节点托管。但它应该驻留在永远在线路由器节点中，以便能够随时接收客户端消息。
* 由于ZigBee的网络或APS层的有效载荷长度较短，因此MQTT-SN消息的最大长度限制在60个字节。