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

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）。
- 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

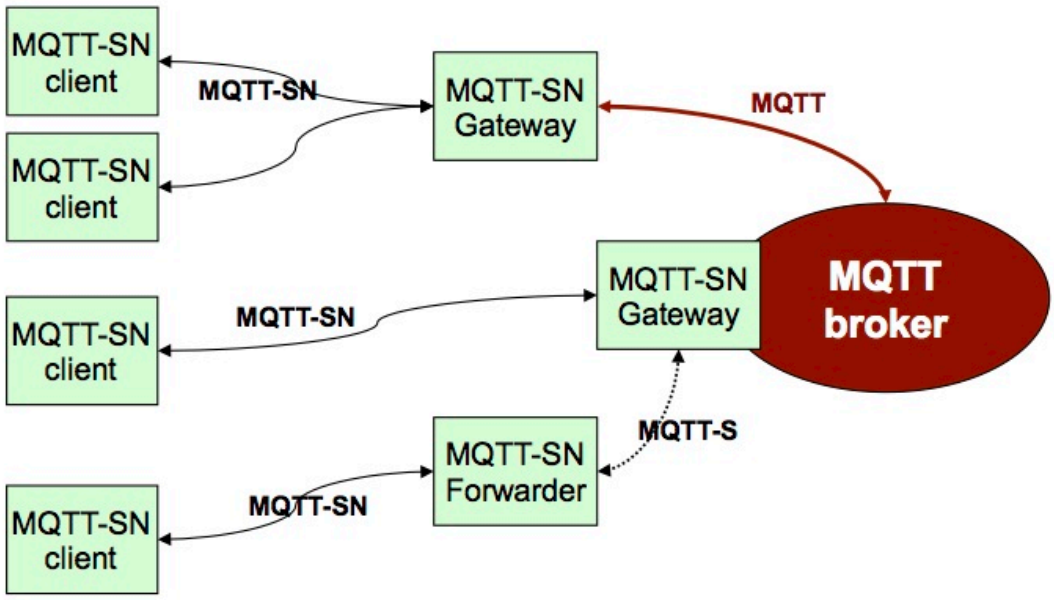

图 1：MQTT-SN 架构

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。

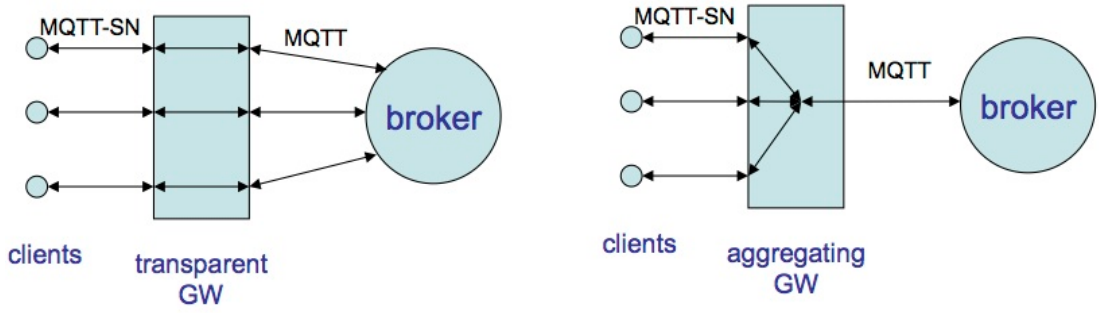

图 2：透传网关和聚合网关

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。

它的格式是如表 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 T_{ADV}，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 N_{ADV} 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 T_{ADV} 应该足够大（例如大于 15 分钟）以避免网络中的带宽拥塞。

T_{ADV} 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 $T_{SEARCHGW}$ 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 R_b 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 R_b 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 R_b 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 T_{GWINFO} 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。

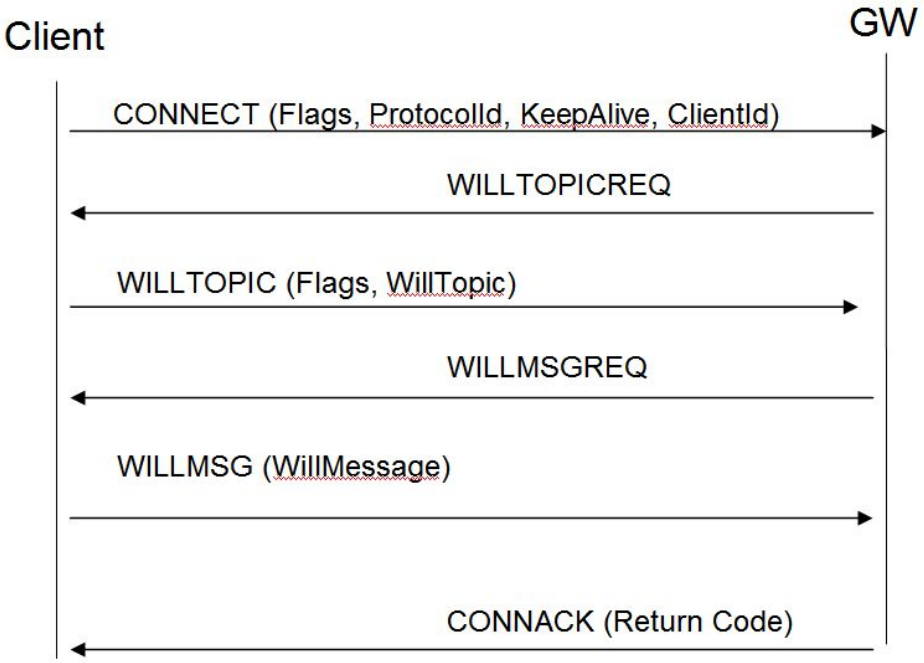

图 3：连接过程

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 T_{WAIT} 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 T_{WAIT} 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 T_{WAIT} 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 T_{retry} 和重试计数器 N_{retry} 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 N_{retry} 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。

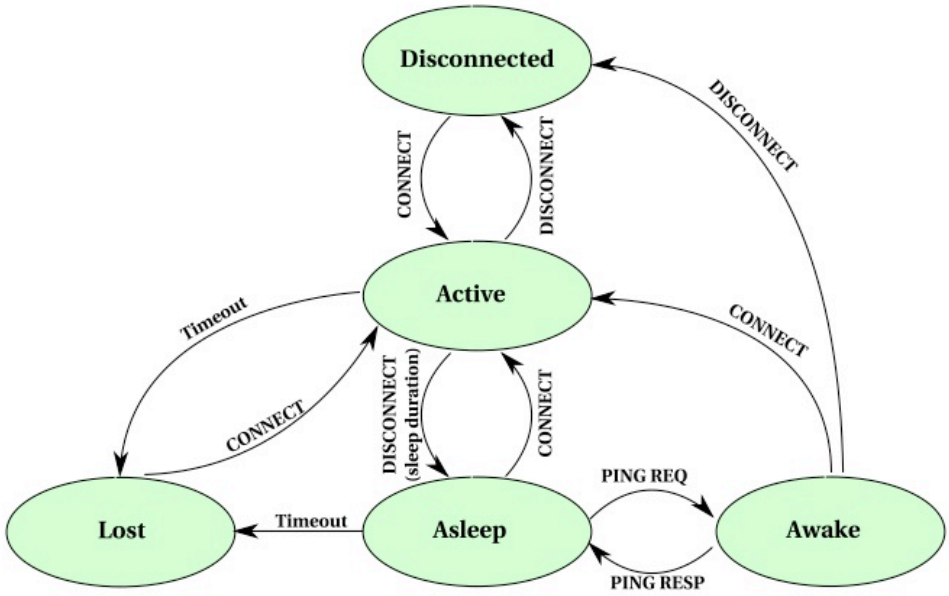

图 4：客户端状态转换

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。

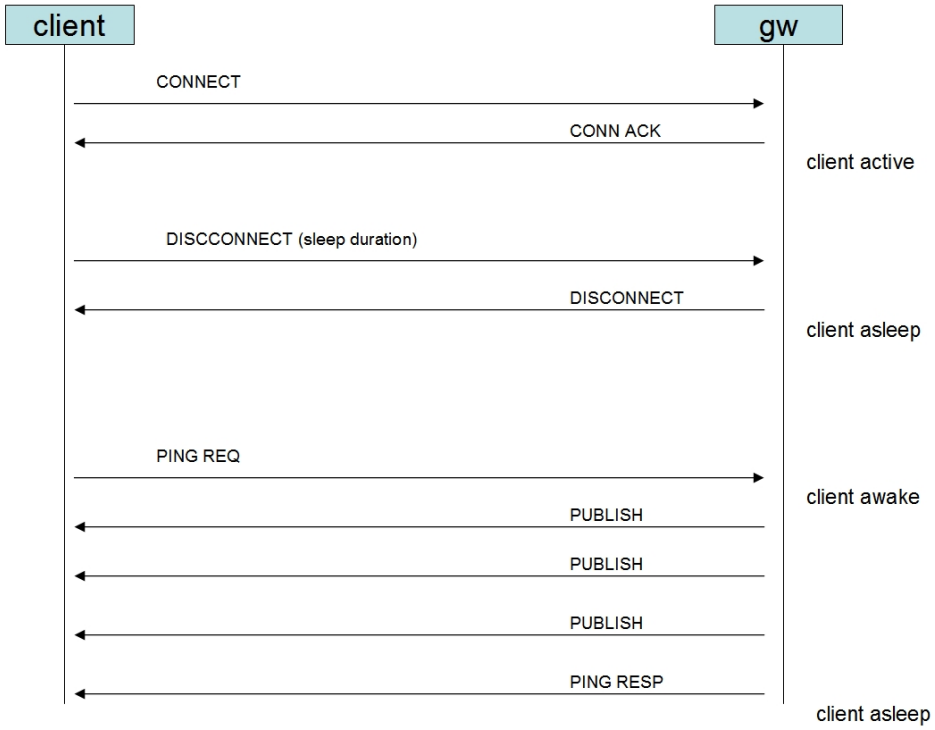

图 5：休眠过程

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
T_{ADV}	大于 15 分钟
N_{ADV}	2-3
$T_{SEARCHGW}$	5 秒
T_{GWINFO}	5 秒
T_{WAIT}	大于 5 分钟
T_{retry}	10-15 秒
N_{retry}	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。