https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/

MQTT, el protocolo de comunicación Machine to Machine (M2M) que tanta popularidad está alcanzando para la comunicación entre dispositivos de IoT.

MQTT son las siglas MQ Telemetry Transport, aunque en primer lugar fue conocido como Message Queing Telemetry Transport.

Es un **protocolo de comunicación M2M (machine-to-machine) de tipo message queue**, **basado en la pila TCP/IP** como base para la comunicación. En el caso de MQTT cada conexión se mantiene abierta y se "reutiliza" en cada comunicación. Es una diferencia, por ejemplo, a una petición HTTP 1.0 donde cada transmisión se realiza a través de conexión.

MQTT fue creado por el Dr. Andy Stanford-Clark de IBM y Arlen Nipper de Arcom (ahora Eurotech) en 1999 como un mecanismo para conectar dispositivos empleados en la industria petrolera.

Aunque inicialmente era un formato propietario, en 2010 fue liberado y pasó a ser un estándar en 2014 según la OASIS (Organization for the Advancement of Structured Information Standards).

**¿CÓMO FUNCIONA EL MQTT?**

El funcionamiento del MQTT es **un servicio de mensajería push con patrón publicador/suscriptor (pub-sub)**. Como vimos en la entrada anterior, en este tipo de infraestructuras los clientes se conectan con un servidor central denominado broker.

|  |  |
| --- | --- |
| Para filtrar los mensajes que son enviados a cada cliente **los mensajes se disponen en topics organizados jerárquicamente**. Un cliente puede publicar un mensaje en un determinado topic. Otros clientes pueden suscribirse a este topic, y el broker le hará llegar los mensajes suscritos. |  |

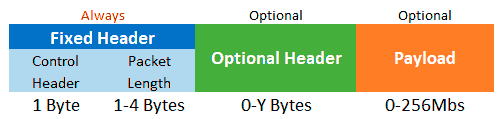
Los clientes inician una conexión TCP/IP con el broker, el cual mantiene un registro de los clientes conectados. Esta conexión se mantiene abierta hasta que el cliente la finaliza. Por defecto, MQTT emplea el puerto 1883 y el 8883 cuando funciona sobre TLS.

|  |  |
| --- | --- |
| El cliente envía un mensaje CONNECT con información necesaria (nombre de usuario, contraseña, client-id…). El broker responde con un mensaje CONNACK, que contiene el resultado de la conexión (aceptada, rechazada, etc). |  |
| Para enviar los mensajes el cliente emplea mensajes PUBLISH, que contienen el topic y el payload. |  |
| Para suscribirse y desuscribirse se emplean mensajes SUBSCRIBE y UNSUSCRIBE, que el servidor responde con SUBACK y UNSUBACK. |  |

Por otro lado, para asegurar que la conexión está activa los clientes mandan periódicamente un mensaje PINGREQ que es respondido por el servidor con un PINGRESP. Finalmente, el cliente se desconecta enviando un mensaje de DISCONNECT.

**ESTRUCTURA DE UN MENSAJE MQTT**

Uno de los componentes más importantes del protocolo MQTT es **la definición y tipología de los mensajes**, ya que son una de las bases de la agilidad en la que radica su fortaleza. Cada mensaje consta de 3 partes:



* **Cabecera fija**. Ocupa 2 a 5 bytes, obligatorio. Consta de un código de control, que identifica el tipo de mensaje enviado, y de la longitud del mensaje. La longitud se codifica en 1 a 4 bytes, de los cuales se emplean los 7 primeros bits, y el último es un bit de continuidad.
* **Cabecera variable**. Opcional, contiene información adicional que es necesaria en ciertos mensajes o situaciones.
* **Contenido(payload)**. Es el contenido real del mensaje. Puede tener un máximo de 256 Mb aunque en implementaciones reales el máximo es de 2 a 4 kB.

Los tipos de mensajes y códigos de control en el protocolo MQTT son los siguientes.

|  |  |
| --- | --- |
| **Message** | **Code** |
| CONNECT | 0x10 |
| CONNACK | 0x20 |
| PUBLISH | 0x30 |
| PUBACK | 0x40 |
| PUBREC | 0x50 |
| PUBREL | 0x60 |
| PUBCOMP | 0x70 |
| SUSBSCRBE | 0x80 |
| SUBACK | 0x90 |
| UNSUSCRIBE | 0xA0 |
| UNSUBACK | 0xB0 |
| PINGREQ | 0xC= |
| PINGRESP | 0xD0 |
| DISCONNECT | 0xE0 |

**CALIDAD DEL SERVICIO (QOS)**

MQTT dispone de un **mecanismo de calidad del servicio o QoS**, entendido como la forma de gestionar la robustez del envío de mensajes al cliente ante fallos (por ejemplo, de conectividad).

MQTT tiene tres niveles QoS posibles.

* **QoS 0 unacknowledged (at most one)**: El mensaje se envía una única vez. En caso de fallo por lo que puede que alguno no se entregue.
* **QoS 1 acknowledged (at least one)**: El mensaje se envía hasta que se garantiza la entrega. En caso de fallo, el suscriptor puede recibir algún mensaje duplicados.
* **QoS 2 assured (exactly one)**. Se garantiza que cada mensaje se entrega al suscriptor, y únicamente una vez.

Usar un nivel u otro **depende de las características y necesidades de fiabilidad de nuestro sistema**. Lógicamente, un nivel de QoS superior requiere un mayor intercambio mayor de mensajes de verificación con el cliente y, por tanto, mayor carga al sistema.

**SEGURIDAD EN MQTT**

La seguridad siempre debe ser un factor importante a considerar en cualquier sistema de comunicación M2M. El protocolo MQTT **dispone de distintas medidas de seguridad** que podemos adoptar para proteger las comunicaciones.

Esto incluye **transporte SSL/TLS y autentificación por usuario y contraseña o mediante certificado**. Sin embargo, hay que tener en cuenta que muchos de los dispositivos IoT disponen de escasa capacidad, por lo que el SLL/TLS puede suponer una carga de proceso importante.

En muchos casos, la autentificación consiste en una contraseña y usuario que son enviados como texto plano. Por último, también es posible configurar el broker para aceptar conexiones anónimas.

Todo esto debe ser tenido en cuenta a la hora de configurar un sistema MQTT, y entender los riesgos de cada uno de ellos, así como su impacto en la eficiencia del sistema.

**VENTAJAS DEL MQTT**

Son varias las ventajas del protocolo MQTT como sistema de comunicación M2M. Por un lado, **tenemos todas las ventajas del patrón pub/sub** que vimos en la entrada anterior, como son escalabilidad, asincronismo, desacomplamiento entre clientes.

Además, MQTT aporta una serie de características que le han hecho sobre salir sobre otros competidores. La principal, como hemos mencionado, **es su sencillez y ligereza**. Esto lo hace adecuado para aplicaciones IoT, donde frecuentemente se emplean **dispositivos de escasa potencia**.

Además, esto menor necesidad de recursos se traduce en **un menor consumo de energía**, lo cual es interesante en dispositivos que funcionan 24/7 y muy especialmente en dispositivos alimentados por batería.

Otra consecuencia de la ligereza del protocolo MQTT es que **requiere un ancho de banda mínimo**, lo cual es importante en redes inalámbricas, o conexiones con posibles problemas de calidad.

Por último, MQTT dispone de medidas adicionales importantes, como la **seguridad y calidad del servicio** (QoS). Por último, es una solución largamente testada y consolidad, que **aporta robustez y fiabilidad**.

**CONCLUSIÓN**

Un protocolo muy interesante para IoT (aunque ya vimos que no es el único). MQTT tiene las ventajas de un patrón pub/sub, y destacando por la sencillez de la comunicación.

El protocolo MQTT se ha alzado como **uno de los estándares para aplicaciones IoT** tanto comerciales como de ámbito maker. Por supuesto, hay muchos más aspectos de los que podríamos hablar mucho más sobre MQTT, como funciones avanzadas de seguridad, permanencia de los mensajes en el broker, configuración de varios brokers.