Introducción al protocolo MQTT

¿Qué es MQTT?

La mejor manera de iniciar esta discusión sobre MQTT es compartir el resumen oficial de la especificación:

“MQTT es un protocolo de transporte de mensajes de publicación/suscripción entre cliente y servidor. Es liviano, abierto, simple y está diseñado para que sea fácil de implementar. Estas características lo hacen ideal para su uso en muchas situaciones, incluidos entornos restringidos como la comunicación en contextos de máquina a máquina (M2M) e Internet de las cosas (IoT) donde se requiere una pequeña huella de código y/o el ancho de banda de la red es escaso”.

Cuando se describe a MQTT como “liviano”, significa que ha sido diseñado para ser simple, eficiente y no consumir muchos recursos. MQTT fue creado con el objetivo de enviar pequeñas cantidades de datos a través de redes poco confiables con ancho de banda y conectividad limitados. En comparación con otros protocolos, MQTT tiene una pequeña huella de código, baja sobrecarga y bajo consumo de energía. Debido a su mínima sobrecarga de paquetes, MQTT se destaca al transferir datos a través de la red en comparación con protocolos como HTTP. Esto también lo hace ideal para su uso en dispositivos con capacidad de procesamiento, memoria y duración de batería limitadas, como sensores y otros dispositivos IoT.

MQTT utiliza un formato de mensaje binario para la comunicación entre clientes y brókeres, a diferencia de otros protocolos que utilizan formatos basados en texto, como HTTP o SMTP.

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type				Flags specific to each MQTT Control Packet type			
byte 2...	Remaining Length							

El formato binario que utiliza MQTT está diseñado para reducir el tamaño de los mensajes y aumentar la eficiencia de la comunicación. Al utilizar un formato binario, el protocolo puede minimizar la cantidad de datos que se deben transmitir y reducir la potencia de procesamiento necesaria para interpretar los mensajes. Esto hace que MQTT sea ideal para su uso en entornos con poco ancho de banda o bajo consumo de energía, como dispositivos IoT con recursos limitados. También se utiliza en sistemas empresariales, donde es necesaria la comunicación de datos en tiempo real.

Otro aspecto importante del protocolo es que MQTT es extremadamente fácil de implementar en el lado del cliente. La facilidad de uso fue una preocupación clave en el desarrollo de MQTT y esto lo convierte en una opción perfecta para dispositivos con limitaciones de recursos.

MQTT se utiliza ampliamente en aplicaciones de IoT, IoT industrial (IIoT) y M2M.

A continuación se muestran algunos ejemplos:

Casas inteligentes : MQTT se utiliza para conectar varios dispositivos en una casa inteligente, incluidos termostatos inteligentes, bombillas, cámaras de seguridad y otros electrodomésticos. Esto permite a los usuarios controlar sus dispositivos domésticos de forma remota mediante una aplicación móvil.

Automatización industrial : MQTT se utiliza para conectar máquinas y sensores en fábricas y otros entornos industriales. Esto

permite el control y monitoreo en tiempo real de los procesos, lo que puede mejorar la eficiencia y reducir el tiempo de inactividad.

Agricultura : MQTT se utiliza en la agricultura de precisión para monitorear los niveles de humedad del suelo, las condiciones climáticas y el crecimiento de los cultivos. Esto ayuda a los agricultores a optimizar el riego y otras prácticas de gestión de cultivos.

Atención sanitaria : MQTT se utiliza para conectar dispositivos médicos y sensores, como medidores de glucosa y monitores de frecuencia cardíaca, a los proveedores de atención sanitaria. Esto permite el seguimiento remoto de los pacientes, lo que puede mejorar los resultados de los pacientes y reducir los costes sanitarios.

Transporte : MQTT se utiliza en automóviles conectados y otros sistemas de transporte para permitir el seguimiento y control de vehículos en tiempo real. Esto puede mejorar la seguridad y ayudar a optimizar el flujo de tráfico.

El origen de MQTT

En 1999, Andy Stanford-Clark de IBM y Arlen Nipper de Arcom (ahora Cirrus Link) desarrollaron el protocolo MQTT para permitir una pérdida mínima de batería y un uso mínimo del ancho de banda al conectarse con oleoductos vía satélite. Los inventores especificaron varios requisitos para el protocolo, entre ellos:

Implementación sencilla

Entrega de datos de calidad del servicio

Ligero y eficiente en el uso del ancho de banda

Datos agnósticos

Conciencia de sesión continua

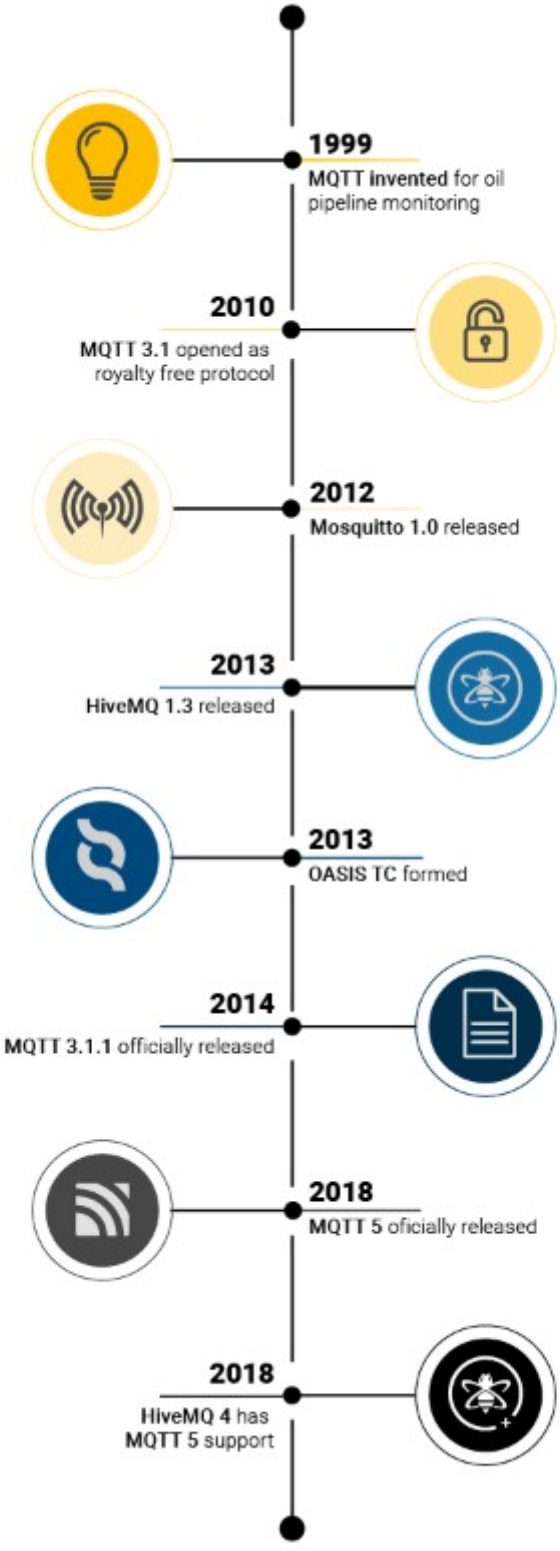
Estos objetivos siguen siendo la base de MQTT. Sin embargo, el enfoque principal del protocolo ha cambiado desde sistemas integrados propietarios a casos de uso abiertos de Internet de las cosas (IoT).

Durante los diez años siguientes, IBM utilizó el protocolo internamente hasta que lanzó MQTT 3.1 como una versión libre de regalías en 2010. Este cambio de enfoque de los sistemas integrados propietarios a los casos de uso abiertos de IoT generó confusión sobre el acrónimo MQTT. Si bien anteriormente significaba MQ Telemetry Transport, donde MQ se refería a MQ Series, un producto desarrollado por IBM para soportar el transporte de telemetría MQ, MQTT ya no es un acrónimo. Ahora es simplemente el nombre del protocolo.

Cuando Andy y Arlen crearon este protocolo en 1999, lo bautizaron como el producto de IBM. Aunque muchas fuentes etiquetan a MQTT como un protocolo de cola de mensajes, esto no es del todo exacto. Si bien es posible poner mensajes en cola en ciertos casos, MQTT no es una solución de cola de mensajes tradicional.

En 2011, IBM contribuyó con implementaciones de clientes MQTT al proyecto Paho de Eclipse Foundation , una corporación independiente sin fines de lucro que ofrece una comunidad para proyectos de software de código abierto. Este fue un avance significativo para el protocolo porque creó un ecosistema más favorable para MQTT. Al contribuir con implementaciones de clientes MQTT a un proyecto de código abierto como Paho, IBM permitió a los desarrolladores acceder al protocolo y crear sus aplicaciones sobre él. Esta medida ayudó a aumentar la visibilidad y la adopción de MQTT entre la comunidad de desarrolladores.

En 2012, HiveMQ se familiarizó con MQTT y creó la primera versión de su software ese mismo año. En 2013, HiveMQ lanzó su software al público.


1999
MQTT invented for oil pipeline monitoring
2010
MQTT 3.1 opened as royalty free protocol
2012
Mosquitto 1.0 released
2013
HiveMQ 1.3 released
2013
OASIS TC formed
2014
MQTT 3.1.1 officially released
2018
MQTT 5 oficially released
2018
HiveMQ 4 has MQTT 5 support

El papel de OASIS en la estandarización de MQTT

En 2014, OASIS anunció que se haría cargo de la estandarización de MQTT, con el objetivo de convertirlo en un protocolo abierto e independiente de los proveedores. Fundada en 1993 como una organización sin fines de lucro, OASIS (Organización para el Avance de Estándares de Información Estructurada) es un consorcio internacional que desarrolla estándares abiertos para Internet y tecnologías relacionadas.

Ha desarrollado numerosos estándares importantes para sectores como la computación en la nube, la seguridad y la IoT, incluidos AMQP, SAML y DocBook. El proceso de estandarización llevó alrededor de un año y el 29 de octubre de 2014, MQTT se convirtió en un estándar OASIS oficialmente aprobado.

La participación de OASIS en MQTT ha sido fundamental para su éxito como protocolo de IoT ampliamente adoptado. Como organización independiente y neutral, OASIS garantiza que el protocolo se mantenga como un estándar abierto que puede ser implementado por cualquier persona sin tarifas de licencia ni restricciones de propiedad.

Además, OASIS ofrece un foro para que la comunidad se reúna y colabore en las mejoras del protocolo, lo que ha resultado en el desarrollo de MQTT 5, la última versión del protocolo con nuevas características para mejorar la confiabilidad y la escalabilidad.

En marzo de 2019, OASIS ratificó la nueva especificación MQTT 5. Esta versión introdujo nuevas características a MQTT que son necesarias para las aplicaciones de IoT implementadas en plataformas en la nube y casos que requieren mayor confiabilidad y manejo de errores para implementar mensajes de misión crítica.

Explorando MQTT: temas, suscripciones, QoS, mensajería persistente y más

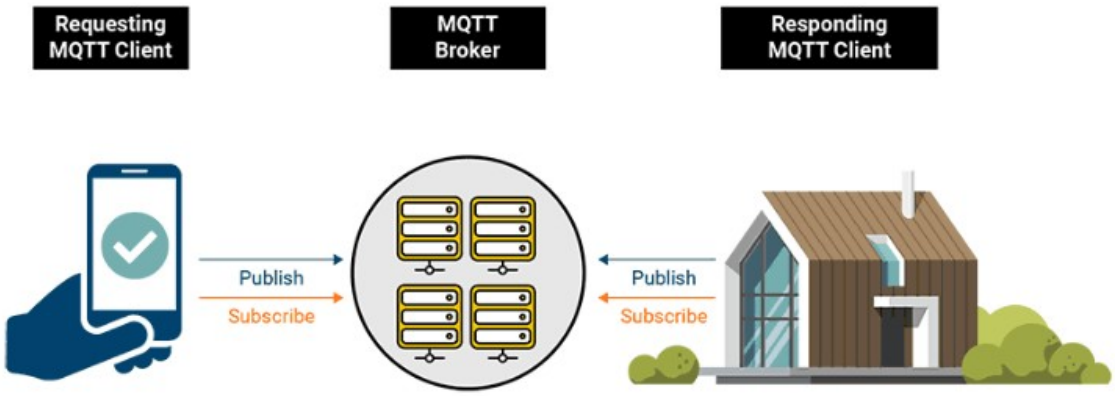
Modelo de mensajería de MQTT: cómo funciona y por qué es importante para IoT e IioT

El modelo de mensajería de MQTT se basa en temas y suscripciones. Los temas son cadenas en las que se publican mensajes y a las que se suscriben. Los temas son jerárquicos y pueden contener varios niveles separados por barras, como una ruta de archivo, como se muestra a continuación.

myhome/kitchen/smartdishwasher

Las suscripciones se utilizan para especificar en qué temas está interesado un cliente en recibir mensajes.

Cuando un cliente se suscribe a un tema, básicamente le está diciendo al broker que está interesado en recibir mensajes publicados en ese tema. El broker luego realiza un seguimiento de la suscripción y reenvía cualquier mensaje publicado en ese tema al cliente suscrito.



Es importante tener en cuenta que un cliente puede suscribirse a varios temas a la vez y que un tema puede tener varios suscriptores. Esto permite contar con un sistema de mensajería flexible y escalable.

Además de los temas y las suscripciones, MQTT también admite comodines, que se pueden utilizar para suscribirse a varios temas que coincidan con un patrón determinado. Los dos tipos de comodines son el comodín de un solo nivel (+), que coincide con un solo nivel de un tema, y el comodín de varios niveles (#), que coincide con todos los niveles posteriores al nivel especificado en un tema.

En general, el modelo de mensajería de MQTT ofrece una forma flexible y escalable de publicar y suscribirse a mensajes mediante temas y suscripciones. El uso de comodines agrega una capa adicional de flexibilidad, lo que permite suscripciones a múltiples temas relacionados mediante una única suscripción.

Comprender el modelo de mensajería de MQTT es crucial, pero igualmente importante es el nivel de calidad de servicio (QoS) que elija para garantizar una entrega confiable de mensajes.

Comprensión de los niveles de calidad de servicio (QoS) de MQTT para aplicaciones de IoT

MQTT admite tres niveles de calidad de servicio (QoS): QoS 0, QoS 1 y QoS 2. A continuación, se muestra el desglose de cada nivel:

QoS 0 : este nivel permite la entrega “como máximo una vez”, en la que los mensajes se envían sin confirmación y pueden perderse. Este es el nivel más bajo de QoS y se utiliza normalmente en situaciones en las que la pérdida de mensajes es aceptable o en las que el mensaje no es crítico. Por ejemplo, QoS 0 puede ser adecuado para enviar datos de sensores en los que la pérdida ocasional de datos no afectaría significativamente los resultados generales.

QoS 1 : este nivel permite la entrega “al menos una vez”, en la que los mensajes se confirman y se vuelven a enviar si es necesario. Con QoS 1, el editor envía el mensaje al intermediario y espera la confirmación antes de continuar. Si el intermediario no responde dentro de un tiempo determinado, el editor vuelve a enviar el mensaje. Este nivel de QoS se utiliza normalmente en situaciones en las que la pérdida de mensajes es inaceptable, pero la duplicación de mensajes es tolerable. Por ejemplo, QoS 1 puede ser adecuado para enviar mensajes de comandos a dispositivos, en los que un comando omitido podría tener consecuencias graves, pero los comandos duplicados no.

QoS 2 : este nivel permite la entrega “exactamente una vez”, en la que los mensajes se confirman y se reenvían hasta que el suscriptor los recibe exactamente una vez. QoS 2 es el nivel más alto de QoS y se utiliza normalmente en situaciones en las que la pérdida o duplicación de mensajes es completamente inaceptable. Con QoS 2, el editor y el intermediario participan en un proceso de confirmación de dos pasos, en el que el intermediario almacena el mensaje hasta que el suscriptor lo recibe y lo reconoce. Este nivel de QoS se utiliza normalmente para mensajes críticos, como transacciones financieras o alertas de emergencia.

Es importante tener en cuenta que los niveles de QoS más altos suelen requerir más recursos y pueden generar mayor latencia y tráfico de red. Por lo tanto, es importante elegir el nivel de QoS adecuado en función de las necesidades específicas de su aplicación.

Además de los tres niveles de Calidad de Servicio, MQTT también admite la persistencia de mensajes, lo que garantiza que los mensajes no se pierdan en caso de una falla de la red o del servidor.

Comprender la persistencia de mensajes MQTT para una comunicación IoT confiable

La persistencia de mensajes es una característica importante de MQTT. Garantiza que los mensajes no se pierdan en caso de que se produzca una falla en la red o en el servidor. En MQTT, la persistencia de mensajes se logra almacenando los mensajes en el servidor hasta que se entregan al suscriptor.

MQTT proporciona tres tipos de opciones de persistencia de mensajes:

No persistente : esta es la opción predeterminada en MQTT. En este modo, los mensajes no se almacenan en el servidor y se pierden si el servidor o la red fallan. Este modo es adecuado para situaciones en las que los mensajes no son críticos y se pueden regenerar fácilmente.

Persistencia en cola : en este modo, los mensajes se almacenan en el servidor hasta que se entregan al suscriptor. Si el suscriptor no está disponible, los mensajes se ponen en cola hasta que el suscriptor se vuelve a conectar. La persistencia en cola es útil cuando el suscriptor no siempre está conectado a la red o si necesita recibir todos los mensajes, incluso si se envían cuando está desconectado.

Persistente con acuse de recibo : este modo proporciona el nivel más alto de persistencia de mensajes. En este modo, los mensajes se almacenan en el servidor hasta que se entregan al suscriptor, y el suscriptor debe confirmar la recepción del mensaje. Si el suscriptor no confirma la recepción, el mensaje se vuelve a enviar hasta que el suscriptor confirme la recepción. Este modo es útil cuando es fundamental garantizar que el suscriptor reciba y procese los mensajes.

Para configurar la persistencia de mensajes en MQTT, el software del broker utilizado para gestionar las conexiones MQTT debe soportar la opción de persistencia elegida. La configuración se

puede realizar a través de los archivos de configuración del broker o a través de su interfaz web.

Es importante tener en cuenta que la persistencia de los mensajes implica una compensación en términos de rendimiento y almacenamiento. Cuanto más persistentes sean los mensajes, más recursos de almacenamiento y procesamiento necesitará el agente. Por lo tanto, es importante elegir el nivel de persistencia adecuado en función de los requisitos específicos de la aplicación.

Seguridad MQTT: Cómo proteger sus dispositivos IoT de los ciberataques

En términos de seguridad, MQTT admite el cifrado TLS para una comunicación segura entre los clientes y el servidor. Existen varias estrategias para proteger las implementaciones de MQTT, como el cifrado de las comunicaciones, la implementación de controles de acceso y autenticación sólidos, etc.

La seguridad de MQTT es un tema complejo que excede el alcance de este artículo. Si está implementando MQTT en su propia aplicación, es importante consultar a un experto en seguridad y seguir las mejores prácticas para proteger su implementación de MQTT.

En general, la arquitectura, el modelo de mensajería y las características de MQTT lo convierten en un protocolo potente y flexible para aplicaciones IoT y M2M. Su diseño liviano y su compatibilidad con niveles de calidad de servicio y persistencia de mensajes lo convierten en una opción ideal para dispositivos y redes con limitaciones.

fuente :HiveMQ, 14 de febrero de 2024
sitio web: https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/