REVISTA CUBANA
DE TRANSFORMACIÓN DIGITAL

RNPS 2487 • ISSN 2708-3411
Vol. 6 • 2025 • e278

ARTÍCULO ORIGINAL

Despliegue de brókers MQTT virtualizados en Docker

Deployment of virtualized MQTT brokers in Docker

Ángel Ramón Coto Rodríguez
angel.ramon@art.jovenclub.cu • ***https://orcid.org/0009-0005-8453-6615***

JOVEN CLUB DE COMPUTACIÓN Y ELECTRÓNICA, DIRECCIÓN PROVINCIAL ARTEMISA, SUBDIRECCIÓN DE INFORMATIZACIÓN

Recibido: 2025-01-23 • Aceptado: 2025-09-05

RESUMEN

Los avances en infraestructuras de las Tecnologías de la Información y las Comunicaciones destinadas al Internet de las Cosas activan todos los sectores de la sociedad contemporánea. Las ventajas que las aplicaciones desarrolladas sobre ellas introducen en el mejoramiento de la calidad de vida de las personas y en el proceso de crecimiento económico hacen pertinente su uso. Tal es el caso de los brókers MQTT, los que permiten conectar dispositivos pequeños y medianos que trabajando de forma autónoma intercambian mensajes para interpretar parámetros físicos o acciones de control. Estos aplicativos de tipo software servidor son libres, con entornos afables y flexibles y resultan sencillos de instalar. En el presente artículo se diseñó una plataforma para Internet de las Cosas utilizando el bróker EMQX y Docker como tecnología de virtualización, estos serán empleados en el despliegue de soluciones tanto en la nube, como en el borde de la red o en redes de área local en dependencia de la aplicación que se desee. Para lograr este objetivo, primeramente, se realizó una revisión detallada sobre los principales conceptos y características de los elementos utilizados, que permitió analizarlos, evaluarlos y elegirlos para el diseño de la plataforma, posteriormente se simularon conexiones y se enviaron mensajes sobre toda la plataforma.

Palabras clave: docker; internet de las cosas IoT; MQTT; plataforma.

ABSTRACT

Advances in Information Technology and Communications infrastructures for the Internet of Things activate all sectors of contemporary society. The advantages that the applications developed on them introduce in the improvement of people's quality of life and in the process of economic growth make their use pertinent. Such is the case of MQTT brokers, which allow connecting small and medium-sized devices that, working autonomously, exchange messages to interpret physical parameters or control actions. These server software type applications are free, with friendly and flexible environments and are easy to install. In this article, a platform for the Internet of Things was designed using the EMQX broker and Docker as virtualization

technology, these will be used in the deployment of solutions both in the cloud, as well as at the edge of the network or in local area networks. depending on the desired application. To achieve this objective, firstly, a detailed review was carried out on the main concepts and characteristics of the elements used, which allowed them to be analyzed, evaluated and chosen for the design of the platform, then connections were simulated and messages were sent over the entire platform.

Keywords: docker; internet of things IoT; MQTT; platform.

INTRODUCCIÓN

La sociedad de los últimos tiempos gira en torno a la transformación digital, y esta, entorno a sus tecnologías habilitadoras. Como parte de ese grupo de ciencias aplicadas se encuentra el Internet de las Cosas (IoT) que puede ser definido como una red de dispositivos que interactúan entre sí (M2M), mediante el Internet (Ray, 2018) y que también se acepta en redes locales. IoT ha permitido que las cosas posean la capacidad de conectarse a la red y se vuelvan inteligentes para crear y obtener datos estratégicos con las mismas que sean aprovechados para minimizar el esfuerzo humano, intercambiar información de forma rápida y en tiempo real, ahorrar tiempo, permitir mayor eficiencia, innovación y productividad de las industrias y obtener ahorro energético.

Con el fin de apoyar la comunicación de miles de millones de dispositivos IoT y rotar las masas de datos producidas, los proveedores de servicios de IoT necesitan implementar y mantener infraestructuras de red robustas y escalables. Cuando las aplicaciones IoT escalan más allá de la escala de las aplicaciones domésticas a mayores sistemas como ciudades o naciones, la cantidad de dispositivos IoT puede crecer muy rápidamente con grados impredecibles. Eso es porque se están desarrollando infraestructuras IoT que no solo tienen una fuerte tolerancia a fallas, sino que además son escalables. La mayoría de las infraestructuras modernas de IoT despliegan un componente esencial conocido como Servidor MQTT (Transporte de telemetría de cola de mensajes). Estos brókers usan el protocolo MQTT, un protocolo abierto Máquina a Máquina (M2M) que existe desde 1999. Gracias a ventajas tales como tamaño compacto, ahorro de ancho de banda, bajo consumo de energía de la batería y separación de espacio/tiempo, los brókers MQTT están dominando cada vez más innegablemente el campo de IoT (Pham, Hoang, & Nguyen, 2021).

El bróker es un elemento que media entre los numerosos dispositivos que generan los mensajes (publicadores) y los suscriptores (clientes que se suscriben a diferentes tópicos), gestionando así las conexiones de red y el intercambio de los mensajes. En algunas circunstancias los mensajes de los publicadores resultan la generación de más datos de IoT de los que se pueden esperar durante tiempos específicos. Esto requiere el desarrollo de nuevos sistemas basados en MQTT que no solo entienden cómo escalar para satisfacer la demanda, sino también para mantenerse al día con los cambios en la carga de trabajo creados por el terminal (Pham, Hoang, & Nguyen, 2021). La nube es una tecnología que sirve de apoyo, por lo que muchos brókers MQTT se despliegan desde la nube en busca de la elasticidad característica que la define, pero al estar demasiado distante de las inmediaciones de las cosas que controlan las acciones, no es recomendable para aplicaciones que requieran rápidos tiempos de respuestas, menor ancho de banda y datos privados, por lo que las arquitecturas actuales de IoT presentan nodos en el borde de la red

que se encargan del procesamiento de las aplicaciones sensibles a la latencia y dejan a la nube para procesar otras, con alta tolerancia al retardo, que requieran de análisis computacionales intensivos y grandes volúmenes de datos.

Este trabajo tiene sus antecedentes en nuestro país en otros artículos donde se han analizado plataformas relacionadas con el Internet de las Cosas (Delgado, 2022) y procedimientos para la implementación de otras tecnologías aplicadas a esta (Pérez, Anías, & Delgado, 2021), así como en una tesis de pregrado de la Universidad de Pinar del Río (Coto, 2022) y en el grupo de investigación del proyecto "Ciudad Inteligente", que ha desarrollado prototipos de sistemas IoT, que no disponían de una plataforma IoT distribuida que empleara bajos recursos y fuera escalable y capaz de brindar alta disponibilidad.

Con estos precedentes y la necesidad de contar con una plataforma IoT, que sea un motor real arquitectónico de soluciones considerable a aplicar en el país, el presente trabajo se centra en la implementación de brókers disponibles gratuitamente en Internet, que soporten el protocolo MQTT, virtualizados éstos sobre contenedores Docker, que pueden ser desplegados tanto en la nube, como en el borde de la red o en redes de área local, en dependencia de las necesidades de la aplicación IoT que se desee.

METODOLOGÍA

La metodología empleada para presentar el diseño de una plataforma IoT de enmarcada aceptación para ser aplicada en el país está compuesta por los cuatro pasos siguientes:

- Paso 1: Análisis de la arquitectura MQTT para IoT a partir de una descripción general.
- Paso 2: Evaluación de las funcionalidades de los principales Brókers MQTT distribuidos que se encuentran disponibles gratuitamente en internet.
- Paso 3: Evaluación de Docker para su uso en la virtualización aplicada al IoT.
- Paso 4: Diseño de la plataforma IoT.

El primer paso se apoya en una revisión de la literatura de los últimos 5 años en Google Académico que apunta a generalidades del protocolo MQTT, que cubren requisitos mínimos necesarios que encajan perfectamente en el desempeño de IoT. El segundo paso comprende una comparativa realizada por el Centro de Investigación Corporativo ABB Alemania, Ladenburg, Alemania (Koziolek, Grüner, & Rückert, 2020) entre tres de los principales brókers MQTT con las mejores prestaciones, lo que permite ofrecer argumentos para decidir por el que más se ajuste al diseño de la plataforma que se hace en el paso 4. En el tercer paso la tecnología de uso prevista está concebida para una solución liviana que sea fácil para el despliegue en entornos distribuidos y cubre ejemplos de escenarios aplicables que fueron presentados en trabajos provenientes de revistas y conferencias publicadas en la biblioteca digital IEEE. Finalmente, con el aporte de los pasos anteriores, en el paso 4 se realiza el diseño de la plataforma IoT que satisface las condiciones de ser económica y eficiente a la hora de aplicarse sobre escenarios reales.

Descripción general de la arquitectura MQTT

MQTT es un protocolo de mensajería de tipo Pub/Sub simple, ligero y basado en TCP/IP (Farhan, Kharel, Kaiwartya, Hammoudeh, & Adebisi, 2018), es capaz de transmitir datos a través de un ancho de banda bajo o redes poco fiables con un consumo muy bajo de poder (OASIS, 2019), esto lo hace aplicable para la comunicación de

dispositivos con recursos limitados, dependientes de baterías y ubicados desde sitios remotos. Las comunicaciones establecidas mediante MQTT presentan baja latencia (OASIS, 2019), lo que convierte al protocolo en ideal para casos en los que se dependa de tratamientos de datos en tiempo real.

Con la intervención de tres elementos de red fundamentales: un publicador (cliente MQTT), un bróker (servidor MQTT) y un suscriptor (cliente MQTT) se realizan comunicaciones bidireccionales. Los publicadores o editores como también suelen llamarse, inician la conexión con el envío de un mensaje hacia el bróker en un asunto determinado. El mensaje que llega al bróker es procesado por este y redirigido hacia todos aquellos clientes que con antelación fueron suscritos al asunto y tienen el interés de recibir el mensaje. Existe la posibilidad de que un cliente se encuentre limitado por el bróker, bien sea para publicar o para suscribirse, así como que requiera una autenticación previa para la conexión. La Figura 1 presenta una arquitectura general de MQTT.

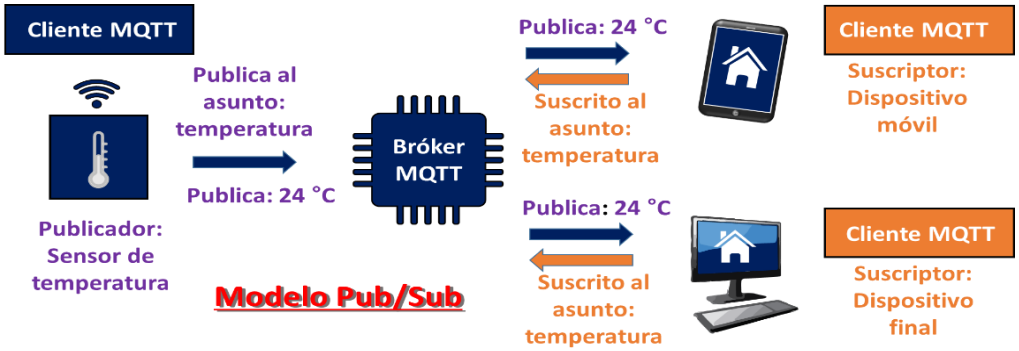

Figura 1. Arquitectura IoT MQTT.

Un mensaje MQTT siempre viajará por la red con una calidad de servicio (QoS) asociada y podrá contener una carga útil de hasta un tamaño máximo de 256 MB (OASIS, 2019). Los nombres de los temas son cadenas codificadas en UTF-8 y se pueden elegir libremente (OASIS, 2019), siempre que no sean restringidos por el bróker. Dado que pueden existir asuntos como parte de estructuras jerárquicas de varios niveles separados por una barra inclinada (/), para el caso de los suscriptores se cuenta con comodines que pueden emplearse para la suscripción de varios asuntos simultáneamente. Por ejemplo, podemos conocer el estado de un sensor de movimiento ubicado en el apartamento de un edifico con el siguiente tema: edificio/piso2/apartamento3/movimiento, pero se pude conocer también el estado de los sensores de movimiento de todos los apartamentos del piso2, empleando el comodín ¨+¨ de la siguiente forma: edificio/piso2/+/movimiento, o el estado de todos los sensores que existan en todos los apartamentos del piso: edificio/piso2/#.

La calidad de servicio (QoS) de MQTT es definida por los clientes, por lo que, al enviarse un mensaje, en el bróker puede existir un punto de cambio de QoS para el reenvío, debido a una elección diferente por parte de quien consumirá el mensaje. El protocolo MQTT tiene tres niveles de QoS, incluidos QoS-0, QoS-1 y QoS-2, que se definen de la siguiente manera: (Archana, Rajeev, Kuruvila, Narayankutty & Kannimoola, 2020)

- QoS-0 (máximo una vez): un paquete se transfiere a un destino hasta una vez.
- QoS-1 (mínimo una vez): cada paquete se emite al destino al menos una vez, esperando que ocurra la iteración del paquete.
- QoS-2 (precisamente una vez): cada paquete se envía a su destino una sola vez.

Brókers MQTT distribuidos

El bróker es un único punto de fallo en el sistema IoT por lo que existen riesgos de perder conexiones si está implementado de manera centralizada, además solo podrá atender una cantidad de clientes específica y procesar una determinada carga de mensajes. En internet se encuentran disponibles gratuitamente varios brókers con características adecuadas para implementaciones en entornos distribuidos, que solucionan este problema y que incluyen asociaciones en puente y en clúster, que según la investigación de la literatura relacionada se presentan como dos de las principales técnicas de diseño para crear una arquitectura distribuida, escalable y de alta disponibilidad. En un modelo en puente los mensajes publicados se reenvían de un intermediario a otro intermediario conectado, a través de reglas establecidas en el puente, lo que garantiza la atención de más mensajes de los clientes. Con un modelo en clúster, varios intermediarios se agrupan formando un único nodo con mayores prestaciones. Los clústeres pueden ser agrupaciones de nodos lógicos o nodos físicos y casi siempre se vinculan con balanceadores de carga que distribuyen las cargas de trabajo entre los elementos del sistema.

Con más de 20 brókers MQTT existentes resulta engorroso realizar una evaluación profunda que compare y de los argumentos suficientes para la decisión de uno u otro bróker, por lo que para nuestra evaluación seleccionamos a EMQX, HiveMQ y VerneMQ, tres de los más empleados en las diferentes soluciones de IoT, que admiten las funcionalidades mencionadas anteriormente y que han sido objeto de estudio del Centro de Investigación Corporativo ABB Alemania, Ladenburg, Alemania, quien los ha evaluado en cuanto a actuación, escalabilidad, disponibilidad/resiliencia, seguridad y extensibilidad, seis principales puntos de decisión con respecto a brókers MQTT que los propios autores han enumerado (Koziolek, Grüner, & Rückert, 2020).

EMQX: Erlang/Enterprise/Elastic MQTT Broker (EMQX) comenzó como un proyecto de código abierto en China en 2013. Los desarrolladores crearon la empresa EMQ Technologies Co., Ltd. en 2017 por soporte y servicios comerciales. La empresa afirma tener más de 5.000 usuarios empresariales y clientes de varios dominios de aplicación. EMQX ahora está disponible en múltiples variantes, como bróker de código abierto puro (1,5 millones de docker pulls), como bróker empresarial y como solución de nube privada. También hay una variante ligera (instalación de 15 MB) llamada "EMQ X Edge" para puertas de enlace de IoT con recursos limitados, que pueden interactuar con KubeEdge. La variante de código abierto está disponible bajo Apache License 2.0 para todos los principales sistemas operativos y arquitecturas de procesador (Koziolek, Grüner, & Rückert, 2020).

HiveMQ: La empresa dc-square inició el desarrollo del bróker comercial MQTT HiveMQ en Alemania en 2012. Se cambió el nombre de dc-square a HiveMQ en 2019 y creó una variante de código abierto (Community Edition, Apache License 2.0, 0,5 millones de docker pulls). La empresa afirma tener más de 130 clientes para HiveMQ, entre ellos BMW con una plataforma de automóvil conectado y Mattenet con una plataforma que proporciona el estado de vuelo en tiempo real de los drones. HiveMQ está implementado en Java y ahora está disponible como comunidad, profesional, y edición empresarial, además de una variante de plataforma en la nube de IoT con Cuota de suscripción. El complemento de detección de DNS de HiveMQ utiliza la detección de servicios de DNS para agregar o eliminar instancias de intermediarios del clúster en tiempo de ejecución (Koziolek, Grüner, & Rückert, 2020).

VerneMQ: Octavo Labs AG de Suiza está desarrollando el VerneMQ Broker MQTT desde 2015. Es un proyecto de código abierto (Licencia Apache 2.0, 7.1M docker pulls) con dos desarrolladores principales que comenzaron después de haber estado trabajando en un proyecto de mercado de la energía. Descubrieron que AMQP y XMPP no escalan lo suficientemente bien para una gran cantidad de dispositivos y comenzaron a implementar VerneMQ utilizando Erlang/OTP. No existen variantes comerciales con derechos de licencia, pero la empresa ofrece apoyo

comercial en torno a VerneMQ. Hay varios clientes destacados, entre ellos Microsoft y Volkswagen (Koziolek, Grüner, & Rückert, 2020).

EQMX mostró el rendimiento más alto con 28 000 msg/s, mientras que VerneMQ logró 10 000 mensajes por segundo y HiveMQ logró 8 000 mensajes por segundo, respectivamente. La escalabilidad de los corredores es potencialmente ilimitada, ya que son de subprocesos múltiples y se pueden escalar horizontalmente. Todos los intermediarios tienen características de seguridad similares y ofrecen extensiones en cualquier lenguaje de programación usando webhooks (Koziolek, Grüner, & Rückert, 2020).

Docker aplicado a IoT

Los escenarios de IoT se han vuelto cada vez más heterogéneos por la disimilitud que existe entre los diferentes nodos implicados. La llegada de las minicomputadoras de placa única como los Raspberry Pi y las placas Odroid, que realizan tareas de procesamiento perimetrales en el perímetro de la red y portan de capacidad de memoria, almacenamiento y conectividad de red, además de acercar la computación a las cosas conectadas han permitido compartir la carga computacional de un sistema con los grandes servidores de los Centros de Datos, complementando así la infraestructura basada en la nube. No obstante, en escenarios IoT específicos puede ser necesario implementar la misma aplicación tanto en el borde de la red como en la nube, esto conlleva a tener que lograr una interoperabilidad entre diferentes sistemas para garantizar que todo permanezca interconectado. La brecha entre IoT, computación en la nube y computación de borde se ha visto reducida con la introducción de las tecnologías de virtualización livianas como la virtualización de contenedores.

El proyecto Docker de código abierto automatiza el despliegue de aplicaciones dentro de contenedores de software, al utilizar imágenes, las que se convierten en contenedores en tiempo de ejecución, optimizando así los recursos de las máquinas virtuales. El entorno Docker proporciona instalaciones para el desarrollo de aplicaciones en el entorno de desarrollo, las pruebas de aplicaciones en el entorno de prueba y la implementación de aplicaciones en el entorno de producción, ya sea en las instalaciones o en la nube (Docker Inc, 2023). La plataforma Docker se utiliza en el entorno de IoT principalmente debido a su ligereza, mejor desarrollo de aplicaciones, implementación rápida de aplicaciones y características de agrupación (Abdul, Mohd Sam, Mohamed, & Dziyauddin, 2019).

De (Gallo, Nguyen, Barone, & van Hien, 2018), los contenedores Docker se utilizan para implementar el intermediario MQTT para el entorno IoT. Mediante el uso de contenedores Docker para el clúster MQTT, los investigadores pudieron implementar un clúster de intermediario MQTT manejable para un mejor rendimiento del intermediario MQTT a bajo costo. Docker también funciona bien en el IoT industrial como se encuentra en (C. A. Garcia, M. V. Garcia, Irisarri, Perez, Marcos, & Estevez, 2018). La plataforma Docker también se usa en la aplicación de control en tiempo real (Tasci, Melcher, & Verl, 2018) mediante el uso del kernel en tiempo real del sistema operativo basado en Linux (Abdul, Mohd Sam, Mohamed, & Dziyauddin, 2019). Ha sido una solución emergente ampliamente aceptada en otras aplicaciones de IoT como los sistemas inteligentes de monitoreo de atención médica (Jaiswal, Sobhanayak, Turuk, Bibhudatta, Mohanta, Jena, 2018), la agricultura de precisión (Marosi, Farkas, & Lovas, 2018) y los hogares inteligentes (Gallo, Nguyen, Barone, & van Hien, 2018).

Diseño de la plataforma IoT

Dadas las potencialidades del bróker EMQX y de Docker antes mencionadas, se consideran estos apropiados para el diseño y despliegue de la plataforma IoT, no obstante, las posibilidades de seleccionar otras herramientas quedan abiertas. Se establecerá comunicación entre las partes implicadas a través del protocolo MQTT. La Figura 2 muestra la plataforma IoT con su arquitectura física, apreciándose la capa inferior con los dispositivos finales

conectados, que mediante un punto de acceso alcanzan un nodo en el borde de la red, que realiza la computación de borde en la capa intermedia y que puede ser cualquier dispositivo con computación, almacenamiento y conectividad de red donde estará ubicado no solo uno, sino dos brókers EMQX agrupados en un clúster. La computación en el borde de la red cercana al usuario final se irá a la computación en nube dispuesta en una capa superior, donde existirá un clúster similar, pero de mayores prestaciones alojado en un servidor con suficiente potencia informática, por mediación de un puente MQTT que se configura solamente en el nodo borde.

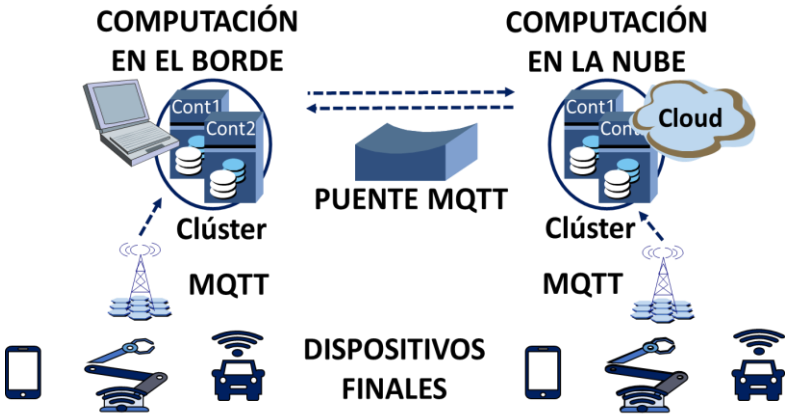

Figura 2. Plataforma IoT (Arquitectura física).

El proceso de configuraciones de la plataforma parte de correr Docker en el host perimetral y en el servidor en la nube, y crear una subred virtual interna que mediante una puerta de enlace enrute los mensajes que lleguen al elemento físico por IP y por el puerto abierto para la escucha de MQTT, hacia una instancia de contenedor con el balanceador de carga HAProxy que garantizará alta disponibilidad de los contenedores del clúster. Para lograr la asociación en clúster se crean y modifican archivos de configuración en los brókers EMQX gracias a la posibilidad de acceso a su código fuente. La Figura 3 presenta la arquitectura lógica de la plataforma.

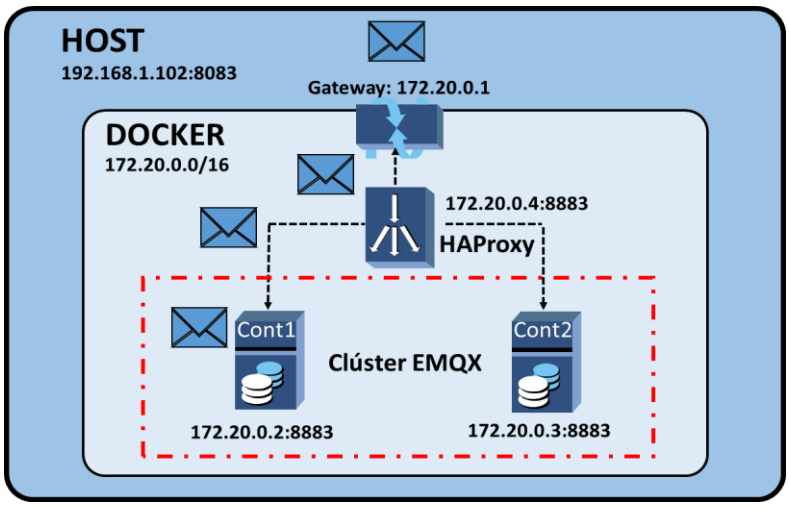

Figura 3. Plataforma IoT (Arquitectura lógica).

RESULTADOS Y DISCUSIÓN

Conexión y envío de datos

Las conexiones al sistema de brókers local (nodo en el borde de la red) y remoto (nodo en la nube) se realizan mediante la creación de clientes en la aplicación de testeo MQTTX y entre ellos a través del puente ya establecido. En el proceso de conexión el balanceador de carga distribuye las solicitudes de forma secuencial entre los nodos del clúster, porque para este caso el algoritmo de programación para HAProxy se ha instalado como estático, aquí se logra la escalabilidad horizontal de la plataforma porque se permiten más conexiones de clientes. Luego de las conexiones, en una interfaz web proporcionada por el bróker EMQX alojado en la nube, se puede comprobar que además de los clientes independientes conectados directamente, existe un cliente conectado mediante el puente que es el nodo de clústeres de alta disponibilidad que está local (bridge_user1). Esto se refleja en la Figura 4.

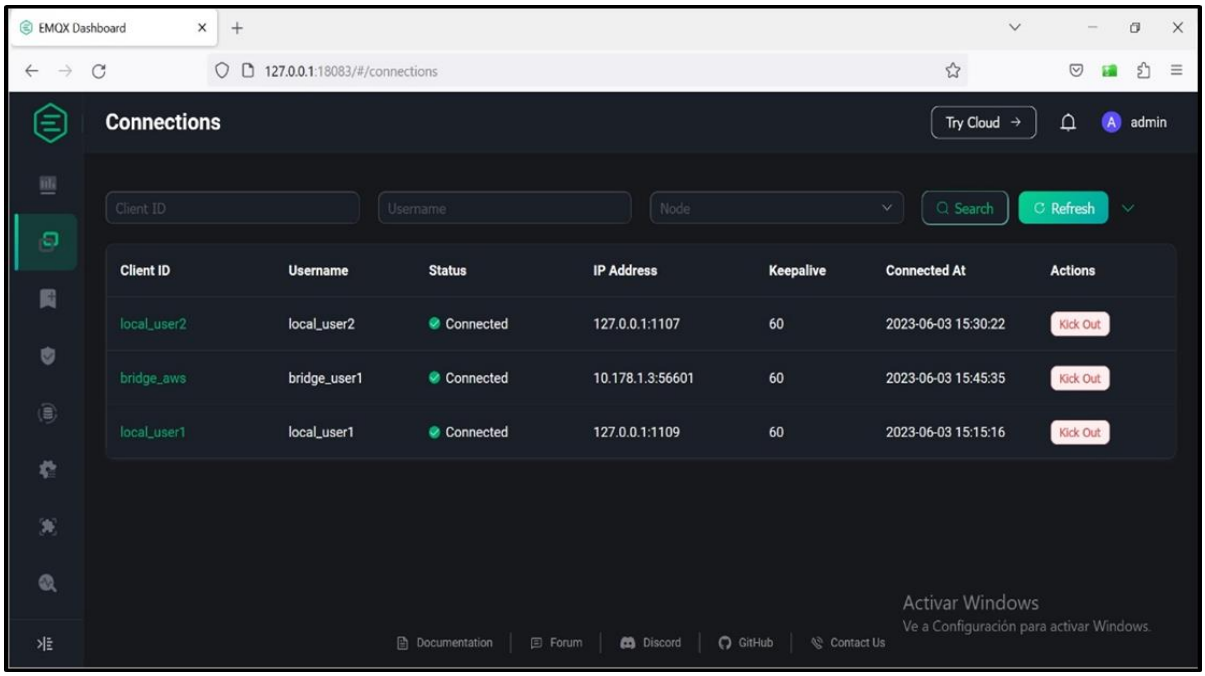

Figura 4. Operación del puente.

Posteriormente comienzan a enviarse datos en asuntos atendidos en cada sistema por separado, incluidos los asuntos específicos declarados en el puente, que llevan el mensaje desde el borde de la red hasta la nube y viceversa. Los clientes de los sistemas están suscritos a todos los tópicos, por lo que recibirán los mensajes en los asuntos actuales y en asuntos futuros que puedan incorporarse, este aspecto puede cambiar dependiendo de las aplicaciones. La Figura 5 muestra mensajes enviados y recibidos entre clientes atendidos por sus respectivos nodos, resaltando los que viajan por el puente. Este punto evidencia la escalabilidad vertical de la plataforma porque el nodo del borde de la red podrá atender más mensajes de los clientes gracias a que una parte de ellos los podrá compartir con la nube.

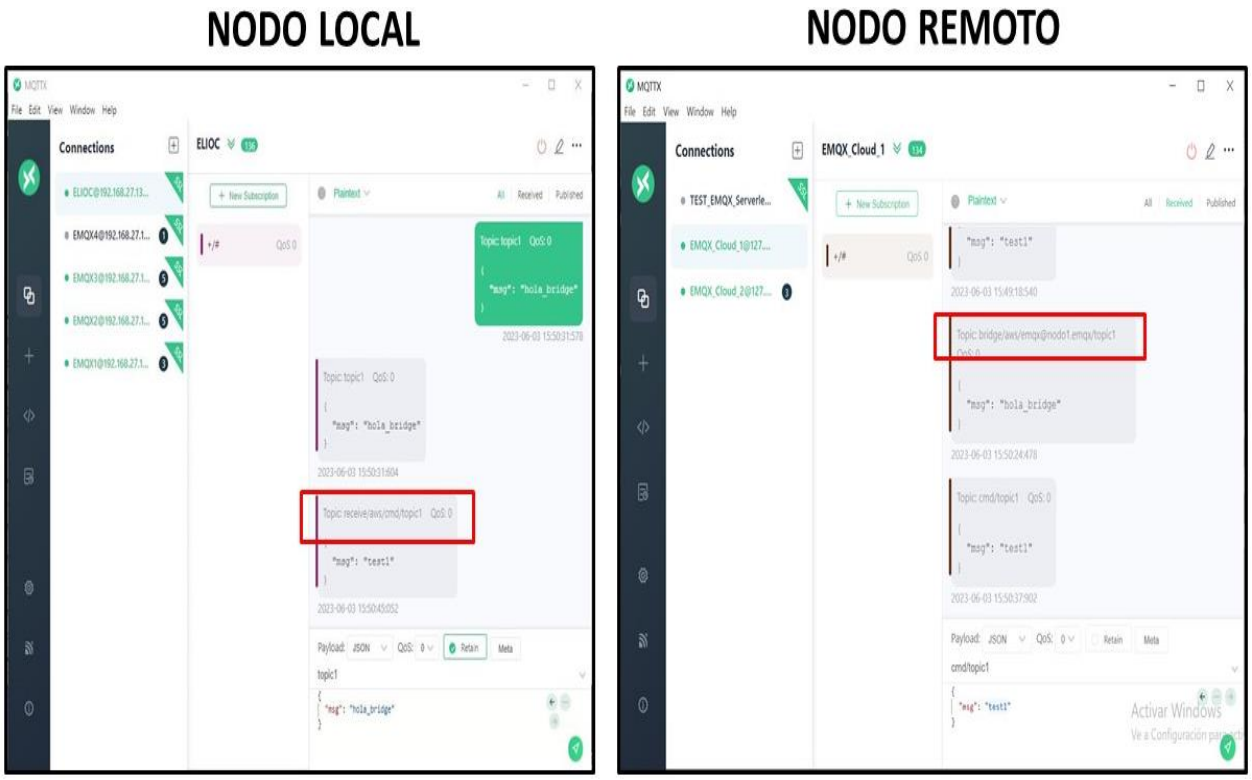

Figura 5. Mensajes enviados y recibidos por los clientes.

Seguridad de la plataforma

En los hallazgos del estudio de (Dizdareviă, Carpio, Jukan, & Masip-Bruin, 2019) se han analizado crecientes ataques de seguridad que se generan para burlar los sistemas IoT y alterar los parámetros provenientes de dispositivos inteligentes que se comunican por protocolos de IoT. Con el fin de contrastar algunas amenazas en la red se incluyen mecanismos de seguridad en toda la plataforma. En las conexiones y envío de datos se usan certificados SSL para ambas direcciones, tanto los servidores como los clientes poseen sus certificados y demuestran que son legítimos, de manera tal que cuando se establecen las comunicaciones ambas partes conocen sus contrapartes, encriptándose la información a nivel de transporte. Todos los clientes han sido registrados con usuarios y contraseñas en las bases de datos internas de los brókers EMQX requiriéndose autenticación previa a nivel de aplicación.

Teniendo en cuenta los resultados anteriores hemos evaluado hasta donde podemos llegar si la plataforma se fuera a desplegar, y es que con el desarrollo en infraestructuras de red del país nuestras ciudades se ven iluminadas con las Redes de Telefonía Celular 3G y 4G, múltiples puntos de acceso a redes WIFI, expansión ADSL y amplia Red Óptica distribuida, por lo que pudiéramos colocar una computación en el borde de la red en cada provincia y subir una nube a los servidores de los Centros de Datos nacionales, lo que nos llevaría a estar en condiciones de atender por ejemplo, los sensores de presión de agua que estuvieran en todas las conductoras del país y seguir un sistema de semejante dimensión, en el que ya en Centros de Datos podemos estar hablando de un gran número de clientes.

CONCLUSIONES

Después de haber realizado este trabajo y teniendo en cuenta los objetivos planteados, se arriba a las siguientes conclusiones:

- La plataforma presentada se aplicó con buenos resultados en el Grupo de Investigación del proyecto "Ciudad Inteligente" de la Universidad de Pinar del Río, como parte de una Tesis de Pregrado.
- Las aplicaciones utilizadas son de software libre, código abierto y posibles de descargar gratuitamente en Internet.
- La instalación de Docker permite el despliegue de clústeres de contenedores de brókers virtualizados, que resultan más eficientes.
- La integración del borde de la red, la nube y el puente, unido a los puntos anteriores, permiten desarrollar soluciones flexibles, escalables, económicas y a la medida de las necesidades de los sistemas IoT, posibles de aplicar en nuestro país en campos como la domótica, la industria, la agricultura, el transporte, la salud, entre otras, para enfrentar muchas dificultades y desafíos existentes.

AGRADECIMIENTOS

El autor desea reconocer la plena dedicación del MSC. Elio Ávila rodríguez desde las primeras etapas de la investigación y le agradece por su valiosa orientación y asesoramiento para alcanzar los resultados presentados.

REFERENCIAS

Abdul, M. S., Mohd Sam, S., Mohamed, N., & Dziyauddin, R. A. (2019). Docker Containers Usage in the Internet of Things: A Survey. Open International Journal of Informatics, 7(Special Issue 2), 208–220. Retrieved from https://oiji.utm.my/index.php/oiji/article/view/95

Archana, E., Rajeev, A., Kuruvila, A., Narayankutty, R., Kannimoola, J.M. (2020). A Formal Modeling Approach for QOS in MQTT Protocol. In: Jain, L., Tsihrintzis, G., Balas, V., Sharma, D. (eds) Data Communication and Networks. Advances in Intelligent Systems and Computing, vol 1049. Springer, Singapore. https://doi.org/10.1007/978-981-15-0132-6_4

Coto, A. R. (2022). Despliegue de clústeres de brokers MQTT virtualizados desde el borde de la red hasta la nube (Tesis de grado). Facultad de Ciencias Técnicas Departamento de Telecomunicaciones y Electrónica, Universidad de Pinar del Río, Cuba.

Delgado, T. (2022). Plataformas IIoT con potencial aplicación en el contexto industrial cubano. Revista Cubana de Transformación Digital, 3(2), 1-12. Retrieved from http://portal.amelica.org/ameli/journal/389/3893437007/

Dizdarević, J., Carpio, F., Jukan, A., & Masip-Bruin, X. (2019). A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration. ACM Computing Surveys, 1(1), 1-30. https://doi.org/0000001.0000001

Docker Inc. (2023). About Docker. Retrieved julio 7, 2023, from Docker: https://www.docker.com/company

Farhan, L., Kharel, R., Kaiwartya, O., Hammoudeh, M., & Adebisi, B. (2018). Towards Green Computing for Internet of Things: Energy Oriented Path and Message Scheduling Approach. Journal Sustainable Cities and Society, 38, 195-204. Retrieved from https://irep.ntu.ac.uk/id/eprint/34328/1/11740_Kaiwartya.pdf

Gallo, P., Nguyen, U. Q., Barone, G., & van Hien, P., "DeCyMo: Decentralized Cyber-Physical System for Monitoring and Controlling Industries and Homes," 2018 IEEE 4th International Forum on Research and Technology for Society and Industry (RTSI), Palermo, Italy, 2018, pp. 1-4, doi: 10.1109/RTSI.2018.8548507.

Garcia, C. A., Garcia, M. V., Irisarri, E., Pérez, F., Marcos, M., & Estevez, E., "Flexible Container Platform Architecture for Industrial Robot Control," 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), Turin, Italy, 2018, pp. 1056-1059, doi: 10.1109/ETFA.2018.8502496.

Jaiswal, K., Sobhanayak, S., Turuk, A. K., Bibhudatta, S. L., Mohanta, B. K., & Jena, D., "An IoT-Cloud Based Smart Healthcare Monitoring System Using Container Based Virtual Environment in Edge Device," 2018 International Conference on Emerging Trends and Innovations In Engineering And Technological Research (ICETIETR), Ernakulam, India, 2018, pp. 1-7, doi: 10.1109/ICETIETR.2018.8529141.

Koziolek, H., Grüner, S., & Rückert, J. (2020). A Comparison of MQTT Brokers for Distributed IoT Edge Computing. Lecture Notes in Computer Science, 12292. doi:https://doi.org/10.1007/978-3-030-58923-3_23

Marosi, A. C., Farkas, A., & Lovas, R., "An Adaptive Cloud-Based IoT Back-end Architecture and Its Applications," 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), Cambridge, UK, 2018, pp. 513-520, doi: 10.1109/PDP2018.2018.00087.

OASIS. (2019). MQTT Version 5.0. Retrieved julio 2, 2023, from OASIS Standard: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

Pérez, A. L., Anías, C., & Delgado, T. (2021). Procedimiento para la implementación de la computación en la niebla en ciudades inteligentes. Revista de Ingeniería Electrónica, Automática y Comunicaciones, 42(1), 45-57. Retrieved from https://www.researchgate.net/publication/351564357

Pham, L. M., Hoang, T.-Q., & Nguyen, X.-T. (2021). Elasticity for MQTT Brokers in IoT Applications. Research and Development on Information and Communication Technology, 2020(2), 62-74. doi:10.32913/mic-ict-research.v2020.n2.941

Ray, P. (2018). A survey on Internet of Things architectures. Journal of King Saud University – Computer and Information Sciences, 30(3), 291-319. Retrieved from https://pdf.sciencedirectassets.com/280416/1-s2.0-S1319157818X00032/1-s2.0-S1319157816300799

Syukor, M., Mohd, S., Mohamed, N., Kamardin, K., & Akmam, R. (2019). Docker Containers Usage in the Internet of Things: A Survey. Open International Journal of Informatics (OIJI), 7(2), 208-220. Retrieved from https://oiji.utm.my/index.php/oiji/article/view/95

Tasci, T., Melcher, J., & Verl, A., "A Container-based Architecture for Real-Time Control Applications," 2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), Stuttgart, Germany, 2018, pp. 1-9, doi: 10.1109/ICE.2018.8436369.