INNOVA UNTREF
Revista Argentina de Ciencia y Tecnología

ICyTEC
Instituto de Ciencia y Tecnología
UNTREF

Seguridad en el envío de mensajes mediante protocolo MQTT en IoT.

Aguirre N., Aranda N., Balich N.

RESUMEN

Internet de las Cosas (IoT) es la interconexión, a través de la red de Internet, de distintos dispositivos que recopilan e intercambian datos de manera automática usando, cada uno, un identificador único. Un componente importante en el sistema de comunicación es el protocolo de comunicación. En IoT se usa generalmente protocolos del estilo publicación / suscripción en donde los proveedores de información no tienen enlace directo con los consumidores de dicha información sino a través de intermediarios, siendo uno de ellos el protocolo MQTT (Transporte de Telemetría de la cola de mensajes) que está construido sobre la pila TCP / IP. Dadas las características de éste protocolo, es posible usarlo tanto en dispositivos industriales como en robots colaborativos, constituyendo el uso de estas dos tecnologías juntas como parte de Internet of Robotic Things (IoRT). De ésta manera IoT permite aumentar la eficiencia y efectividad de diferentes proceso en los que participa desde aquellas de la vida diaria hasta aquellas de alta complejidad y requerimientos de seguridad y en la conjunción con la robótica tiende a desempeñar un papel importante al permitir que los operadores de robots ejerzan un mayor control sobre sus equipos, ya que permitirá la manipulación a distancia utilizando solamente una aplicación, incluso en un entorno cambiante y dinámico ideal para la operación de robots colaborativos.

La rápida evolución de los servicios de IoT plantea también otros problemas como lo es la seguridad, ya que el protocolo MQTT se evidencia como muy vulnerable a posibles ataques representando, de esa manera, un peligro para la integridad de los datos y constituyendo un riesgo para los usuarios. Por lo tanto se hace necesario adoptar enfoques de seguridad que sean garantía de un conjunto de criterios para evitar posibles ataques cibernéticos. Para tal fin, éste trabajo plantea aplicar una capa de seguridad en el contenido de los mensajes que se envían a fin de neutralizar los diferentes medios de intercepción utilizados por los atacantes. Para ello se implementa el protocolo MQTT usando como intermediario un servicio bróker IoT, una aplicación de conexión, una interfaz gráfica y se utiliza mecanismo de cifrado simétrico para la generación de llaves tanto en el cliente emisor como en los dispositivos conectados. El beneficio que aporta este enfoque es la transmisión de mensajes de forma segura y a bajo costo. Hay mucho aún por investigar y desarrollar, en este aspecto, ya que así cómo evoluciona la tecnología, también surgen nuevas formas de producir ataques a datos y dispositivos.

ABSTRACT

Internet of Things (IoT) is the interconnection, through the Internet network, of different devices that collect and exchange data automatically using, each, a unique identifier. An important component in the communication system is the communication protocol. In IoT, protocols of the publication / subscription style are generally used where the information providers have no direct link with the consumers of said information but through intermediaries, one of them being the MQTT (Telemetry Transport of the message queue) protocol. which is built on the TCP / IP stack. Given the characteristics of this protocol, it is possible to use it in both industrial devices and collaborative robots, constituting the use of these two technologies together as part of the Internet of Robotic Things (IoRT). In this way IoT allows to increase the efficiency and effectiveness of different processes in which it participates from those of daily life to those of high complexity and security requirements and in conjunction with robotics it tends to play an important role by allowing operators of robots exercise greater control over their equipment, since it will allow remote manipulation using only one application, even in a changing and dynamic environment ideal for the operation of collaborative robots.

The rapid evolution of IoT services also poses other problems such as security, since the MQTT protocol is shown to be very vulnerable to possible attacks, thus representing a danger to data integrity and constituting a risk to the users. Therefore, it is necessary to adopt security approaches that guarantee a set of criteria to avoid possible cyber attacks. To this end, this work proposes to apply encryption in the content of the messages that are sent in order to neutralize the different means of interception used by the attackers. For this, the MQTT protocol is implemented using as an intermediary an IoT broker service, a connection application and a graphical interface and a symmetric encryption mechanism is used for the generation of keys both in the sending client and in the connected devices. The benefit of this approach is the transmission of messages safely and at a low cost. There is much still to investigate and develop, in this aspect, since this is how technology evolves, new ways of producing attacks on data and devices also emerge.

PALABRAS CLAVE

Seguridad, IoT, MQTT.

KEY WORDS

Security, IoT,MQTT.

INTRODUCCIÓN

El crecimiento acelerado de la población mundial consecuentemente con la necesidad de satisfacer las necesidades de toda la población mundial ha ido acompañado del crecimiento de la industria, el empleo y la interacción hombre - máquina a través de la robótica. A partir del nacimiento de dicha interacción se aumentó una flexibilidad inherente a los procesos de fabricación de la Industria que requería de robots con nuevas capacidades que interactuaran con su entorno, con el propio producto fabricado y con las personas. En consecuencia, se crea un desafío para el futuro en donde los robots y las personas colaborarán para aprovechar lo mejor de los dos mundos, la flexibilidad de las personas y la potencia y precisión de los robots. Pero esta colaboración solo tendrá lugar si es posible garantizar la seguridad de las personas que comparten el lugar de trabajo con los robots. Además, nuevos paradigmas de programación de robots contribuirán a reducir el esfuerzo relacionado con la realización de nuevas tareas. Estas mejoras permitirán que los robots se utilicen en empresas que no los utilizaban debido a su falta de flexibilidad y al esfuerzo de programación que requerían. En robótica el Comité de Normalización ISO TC 184:2016 elabora estándares aplicables a los sistemas de automatización y su integración para el diseño, suministro, fabricación, entrega, mantenimiento y desecho de productos cubriendo áreas como los sistemas de información, el software de automatización y control y las tecnologías de integración, mientras que el comité ISO TC 299: 2016 tiene la responsabilidad de los estándares utilizados en los robots de manipulación controlados automáticamente y reprogramables, tanto fijos como móviles.

Asimismo se ha visto en los últimos tiempos una gran aceleración en las tecnologías de internet como computación en nubes Cloud Computing, almacenamiento en las nubes y otros, que brindan beneficios de la estructura convergente y servicios compartidos a través de una red que usualmente es internet. Es así que surge el concepto de Internet de las Cosas (IoT) y que se aplica a la interconexión digital de objetos cotidianos con internet de manera alternativa. Internet de las Cosas según ([1], Chen, Xu, Liu, Hu, & Wang, 2014), se trata de una red inteligente que permite el intercambio de información y comunicación entre dispositivos inteligentes. Ésta tecnología ha evolucionado con la finalidad de unir un gran número de elementos, con diferentes capacidades de conexión, a una gran y única red, siendo éstos capaces de enviar información sobre la internet, detectar el estado de un entorno y procesar los datos y enviar los resultados ([2] Alvear Puertas, 2017). Otros autores como ([3] Jaehak Byun, 2016) sostienen que el concepto de Internet de las Cosas (IoT) nace como un término para definir un escenarios en los que la conectividad de red y la capacidad de cómputo se extienden a objetos, sensores y artículos de uso diario, permitiendo que estos dispositivos generen, intercambien y consuman datos con una mínima intervención humana. De esta manera IoT permite aumentar la eficiencia y efectividad de diferentes procesos en los que participa desde aquellas de la vida diaria hasta aquellas de alta complejidad y de requerimientos de seguridad.

Las implementaciones de la IoT utilizan diferentes modelos de conectividad, cada uno de los cuales tiene sus propias características. Es por ello que la Junta de Arquitectura de Internet describe cuatro diferentes modelos de conectividad, cada uno con sus propias características. Éstos modelos son: Device-to-Device (dispositivo a dispositivo), Device-to-Cloud (dispositivo a la nube), Device-to-Gateway (dispositivo a puerta de enlace) y Back-End Data-Sharing (intercambio de datos a través del back-end). Estos modelos destacan la flexibilidad en las formas en que los dispositivos de la IoT pueden conectarse y proporcionar un valor para el usuario ([4] Karen Rose, 2015).

El rápido crecimiento de dispositivos conectados nos muestra que la interacción más frecuente con Internet proviene de la interacción pasiva con objetos conectados y no de una interacción activa con el contenido. Éste resultado es una prueba de la naturaleza de propósito general de la propia arquitectura de Internet, que no impone limitaciones inherentes a las aplicaciones o servicios que pueden hacer uso de la tecnología ([5] Hyewon Jeong, 2017).

De esta manera adquiere relevancia el sistema de comunicación y en éste a su vez, el protocolo de comunicación. En IoT generalmente se usan protocolos del estilo publicación / suscripción que son un estilo de mensajería en donde los proveedores de información (los publicadores) no tienen enlace directo con los consumidores de esa información (los suscriptores) ya que este enlace está controlado por un intermediario (bróker). En un sistema de estas características, un publicador no necesita saber quien utiliza la información, de la misma manera que un suscriptor no necesita saber quien proporciona la información que recibe. Solo se requiere que los dispositivos se conecten a un “tópico” de un gestor intermediario y publiquen la información. Los consumidores se pueden conectar al gestor y suscribirse a los datos del tópico. Por ejemplo, un dispositivo puede medir la temperatura cada minuto y publicarlo una vez por hora. Una aplicación suscripta a dicha información recibirá una vez por hora un compendio horario de las muestras tomadas cada minuto. Este modelo desacopla al productor de datos del consumidor de datos. Las publicaciones se envían desde los publicadores al bróker, y éste las reenvía al suscriptor. Es el bróker quien garantiza que los mensajes sean entregados a los suscriptores correctos. El publicador genera el mensaje que desea publicar y define el tema del mensaje, mientras que el suscriptor registra una solicitud para una publicación especificando los temas de los mensajes en los que está interesado. Además puede tener más de un publicador y más de un suscriptor a la vez, de la misma manera que una aplicación puede ser tanto un publicador como un suscriptor. Los sistemas de estas características son más escalables porque el separar a productores de consumidores permite que cada uno se agregue y se quite de forma independiente lo cual hace también pueda tener cuestiones de compatibilidad por el mismo motivo.

En éste estilo de sistema de publicación / suscripción se destaca el protocolo MQTT (Message Queue Telemetry Transport), creado por IBM y liberado enfocando la conectividad Machine-to-Machine (M2M). Es un protocolo de mensajería que soporta la comunicación asíncrona entre las partes, por lo tanto disocia al emisor y al receptor de

los mensajes tanto en espacio como en tiempo, siendo escalable en entornos de red no confiables. Éste protocolo de transporte de mensajes está basado en publicaciones y subscripciones a los denominados “tópicos” o sea cada vez que un mensaje es publicado será recibido por el resto de dispositivos adheridos a un tópico del protocolo.

Dada sus características de que necesita de muy pocos recursos para su funcionamiento hacen que éste protocolo sea ideal para el mundo emergente de "máquina a máquina" (M2M) o "Internet de las cosas" de dispositivos conectados, como así también para aplicaciones móviles donde el ancho de banda y la potencia de la batería son de primera. Esto ha hecho que rápidamente se convierta en un protocolo muy empleado en la comunicación de sensores y por consiguiente, dentro del Internet de las Cosas.

El protocolo MQTT funciona sobre TCP/IP o sobre otros protocolos de red con soporte bi-direccional y sin pérdidas de datos, teniendo como principal característica el uso de mensajes “broadcast” para subscripción y publicación de datos con independencia de la aplicación. En cuanto a su arquitectura, sigue una topología en estrella, donde existe un nodo central o broker con capacidad para trabajar con un gran número de clientes y también encargado de gestionar la red y transmitir los mensajes.

Algunas ventajas del Protocolo MQTT se deben a que es un protocolo de red abierto, sencillo, ligero y fácil de implementar, ideal para responder a muchas necesidades y especialmente adaptado para utilizar un ancho de banda mínimo. Otras ventajas del MQTT radican en que su flexibilidad hace que pueda soportar varios escenarios de aplicaciones para dispositivos y servicios de IoT ([6] Franco, 2015).

Algunas desventajas del protocolo están vinculadas al patrón de solicitud y respuesta cuando un dispositivo envía sus datos como una solicitud y recibe actualizaciones del sistema como la respuesta. Esto dado que el cliente debe iniciar una conexión, por lo cual en la aplicación de IoT los dispositivos y los sensores son normalmente los clientes, lo que significa que no pueden recibir comandos de la red de forma pasiva.

La tendencia a que haya una hiperconectividad da origen uno de los retos actuales de la tecnología de la información y es la ciberseguridad. En este aspecto hay un término que ha revolucionado este ámbito, se trata del blockchain, un sistema que permitirá a las empresas generar confianza respecto al intercambio de información ya que las transacciones tienden a ser seguras siendo una oportunidad para que las empresas puedan acogerse. Una característica interesante es la capacidad de MQTT para establecer comunicaciones cifradas lo que aporta a la red una capa extra de seguridad.

OBJETIVOS

Consecuentemente con la evolución que se registra en IoT, ya que se está observando un veloz crecimiento en cantidad de dispositivos conectados a la red mundial también se plantea el problema de la seguridad y privacidad en lo referido a los dispositivos y a los datos lo que implicaría incluir controles criptográficos, controles de acceso, controles de flujo de información, controles de inferencia y procedimientos para respaldo y seguridad.

Los sistemas IoT son susceptibles de amenazas permanentes, lo cual implica nuevos desafíos para desarrollar protección no solo de dispositivos y datos sino también de claves criptográfica y de credenciales ([7] Perez, 2018). A medida que se intensifican los intentos por explotar fallas en la seguridad, también se intensifica el amplio espectro de vulnerabilidades. En las específicas de la comunicación, los atacantes intentan, a menudo, de diversas formas interceptar, interrumpir o falsificar los mensajes enviados desde los dispositivos.

Si tenemos en cuenta además, que uno de los principios fundamentales de la seguridad es proporcionar confidencialidad, integridad y disponibilidad de los recursos críticos, entonces uno de los grandes desafíos que se impone IoT es encontrar un entorno de comunicación confiable que garantice la seguridad de los datos transmitidos entre todos los dispositivos interconectados y lograr que éstos sean lo menos vulnerables posible. En la actualidad no se puede garantizar el entorno deseado de una manera segura, aunque muchas empresas y organizaciones trabajan para mejorar cada vez más esta cobertura de Internet de las Cosas, aun no es posible evitar los ataques, pero si hacerlos mucho más difícil ([8] Eterovic, 2019).

El objetivo de este trabajo es lograr seguridad en los mensajes enviados utilizando el protocolo MQTT, hacia un dispositivo o robot y aplicando una capa de seguridad mediante encriptación de cifrado simétrico en el contenido de los mismos. Se intenta de ésta manera realizar un nuevo aporte con un enfoque distinto a los trabajos de investigación que ya existen sobre la seguridad en el protocolo MQTT en IoT.

Los lineamientos generales de la presente investigación es mostrar los medios y recursos utilizados, tanto en hardware como en software con su respectiva arquitectura, luego los métodos aplicados de intercepción de mensajes y finalmente exhibir la solución planteada para lograr a los resultados deseados.

METODOLOGÍA

Para lograr el objetivo de éste trabajo se propone es realizar la comunicación con un robot utilizando un protocolo del estilo publicación / suscripción, un elemento de IoT, como lo es el protocolo Mqtt para luego producir una intercepción al mensaje durante el envío

hacia el robot, mostrando la vulnerabilidad expuesta y finalmente brindar la solución propuesta

El desarrollo se realiza utilizando como hardware, un kit básico de robot móvil con dos placas interconectadas entre sí. Para la interconexión entre las dos placas se utiliza tecnología I2C. Para la comunicación entre el robot y el usuario se utiliza el servicio Mqtt del software Mosquitto de Eclipse que realiza las tareas de bróker en el manejo de la publicación y suscripción de los mensajes. Como complemento y con el objetivo de otorgar mayor versatibilidad en el desplazamiento del robot, se le agrega un sensor ultrasonido para evitar obstáculos que se puedan presentar durante la marcha. Éste mismo esquema también se puede utilizar en otros tipos de equipos o dispositivos, como por ejemplo robots bípedos trabajando en un entorno dinámico y cambiante.

Las placas utilizadas son una placa Wemos D1 y una placa Arduino Uno. La primera en su función de Master es la que se encarga de la conexión a internet, mientras que la segunda, con la función de Slave es la que se encarga del resto de las tareas que se le asignan al robot como ser la movilidad y los dispositivos utilizados para sensar el ambiente dinámico y cambiante en el que trabaja. Se utiliza también un sensor de ultrasonido que trabaja en el robot a fin de evitar los posibles obstáculos que se presenten durante la marcha del robot.

En cuanto al software, para las placas Arduino Uno y Wemos D1 se utilizó software Arduino con sus correspondientes librerías. Para conexión entre ambas placas se realizó con tecnología I2C utilizándose también sus correspondientes librerías.

Inter Integrated Circuit (I2C) como bus serie de datos fue desarrollado originalmente para la comunicación entre diferentes partes de un circuito. Diseñado como bus maestro-esclavo en donde la transferencia de datos es siempre inicializada por el maestro, haciendo que reaccione el esclavo. Es un protocolo síncrono que para conectar dos placas entre si y usa un cable de conexión para el reloj (SCL) y otro para el dato (SDA). El maestro y el esclavo envían datos por el mismo cable pero que es controlado por el maestro que crea la señal de reloj. El inicio de la transmisión es indicado por la señal de inicio del maestro seguido de la dirección. Ésta es confirmada por el ACK Bit del esclavo. El ACK es enviado desde el esclavo al escribir o desde el maestro al leer. El último Byte leído es reconocido por el maestro como un NACK (not acknowledge) para indicar el final de la transmisión y es finalizada por la señal de la parada.

Los clientes MQTT pueden ser dispositivos conectados con sensores y sistemas embebidos o aplicaciones ejecutando alguna librería MQTT y que de alguna forma interactúen con los datos.

Los publicadores y suscriptores de mensajes pueden controlar y configurar los sensores a su cargo mediante comandos, si es que son nodos de sensores.

El bróker es un software que brinda el servicio, implementa el protocolo MQTT, establece comunicación a nivel de aplicación entre los diferentes clientes, autoriza el acceso, los identifica, y es además responsable de recibir los mensajes, filtrarlos y rutearlos a los clientes suscriptos según su topic.

El topic es el tema o categoría del mensaje, funcionando como un identificador que define el contenido del mensaje o su ámbito de influencia produciendo una categorización jerárquica tipo árbol.

Para publicar un mensaje MQTT, se debe clasificar en uno o varios topics concreto según se haya suscripto, y los clientes podrán recibir los mensajes respectivos.

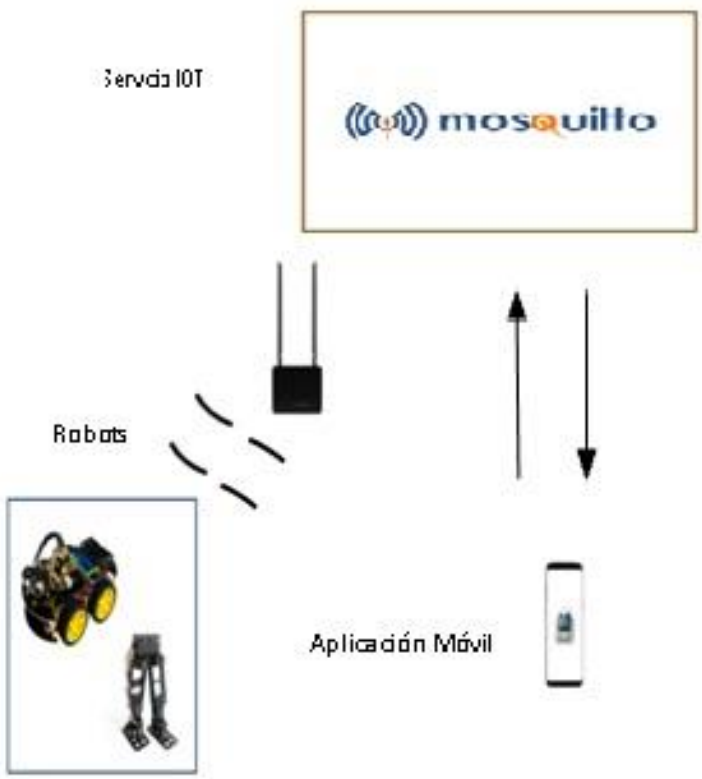
El topic string es una cadena de caracteres que identifica el topic final del mensaje y puede ser multinivel, que utiliza el carácter (wildcard) # para admitir cualquier nivel por debajo del nivel mostrado en la cadena o simple que utiliza el carácter (wildcard) + para admitir sólo hasta el nivel mostrado en la cadena.

Como broker se utilizó el software Mosquitto de Eclipse mediante su servicio Mqtt. También se desarrolla una interfaz gráfica que permite el manejo a nivel de usuario, y que se conecta con el software que opera el servicio del protocolo Mqtt. Para ello, se crea una aplicación móvil desarrollada con Apache Cordova que actúa para enviar y recibir las acciones a través del servicio Mqtt de Mosquitto hacia el dispositivo.

La intercepción al mensaje se realiza mediante Wireshark y de esta manera se pueda verificar la vulnerabilidad a que la que queda expuesto el mensaje enviado.

En la figura 1 se puede apreciar la arquitectura descripta.

Figura 1: Arquitectura utilizada.



Fuente: Elaboración propia.

En la figura 2 muestra las interfaces de la aplicación.

Figura 2: Interfaces de la aplicación móvil.

Fuente: Elaboración propia.

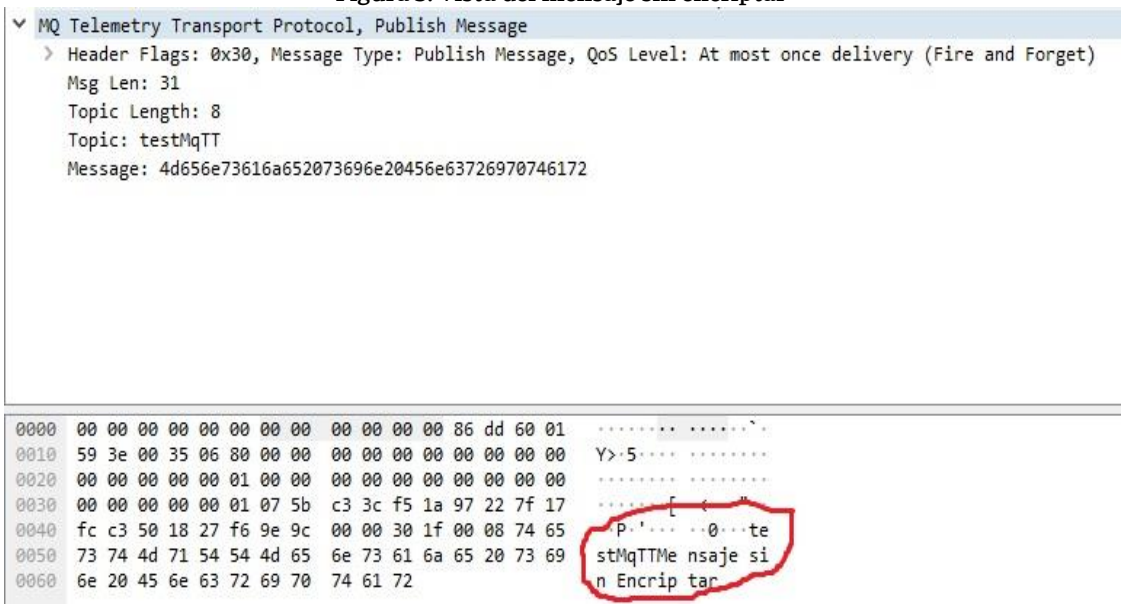
Estas interfaces permiten configurar el servidor Mqtt y también recibir y/o enviar instrucciones a los robots que se encuentran suscriptos.

RESULTADOS Y DISCUSIÓN

En la primera parte de esta etapa se va a observar la vulnerabilidad en el sistema sin seguridad al realizar un ataque de intercepción al mensaje. Para realizar esta tarea se utiliza la herramienta Wireshark.

En la figura 3 se puede observar un mensaje interceptado y una lectura completa del mismo que está marcado abajo en el extremo derecho de la imagen.

Figura 3: Vista del mensaje sin encriptar



Fuente: Elaboración propia.

Cifrar el mensaje de MQTT, garantiza que los mensajes enviados no podrán ser manipulado o leído sin tener la clave para descifrarlo. En el caso que se intervenga y espié la comunicación, solo se podrá observar un mensaje incomprensible ya que tiene una implementación de criptografía autenticada simétrica.

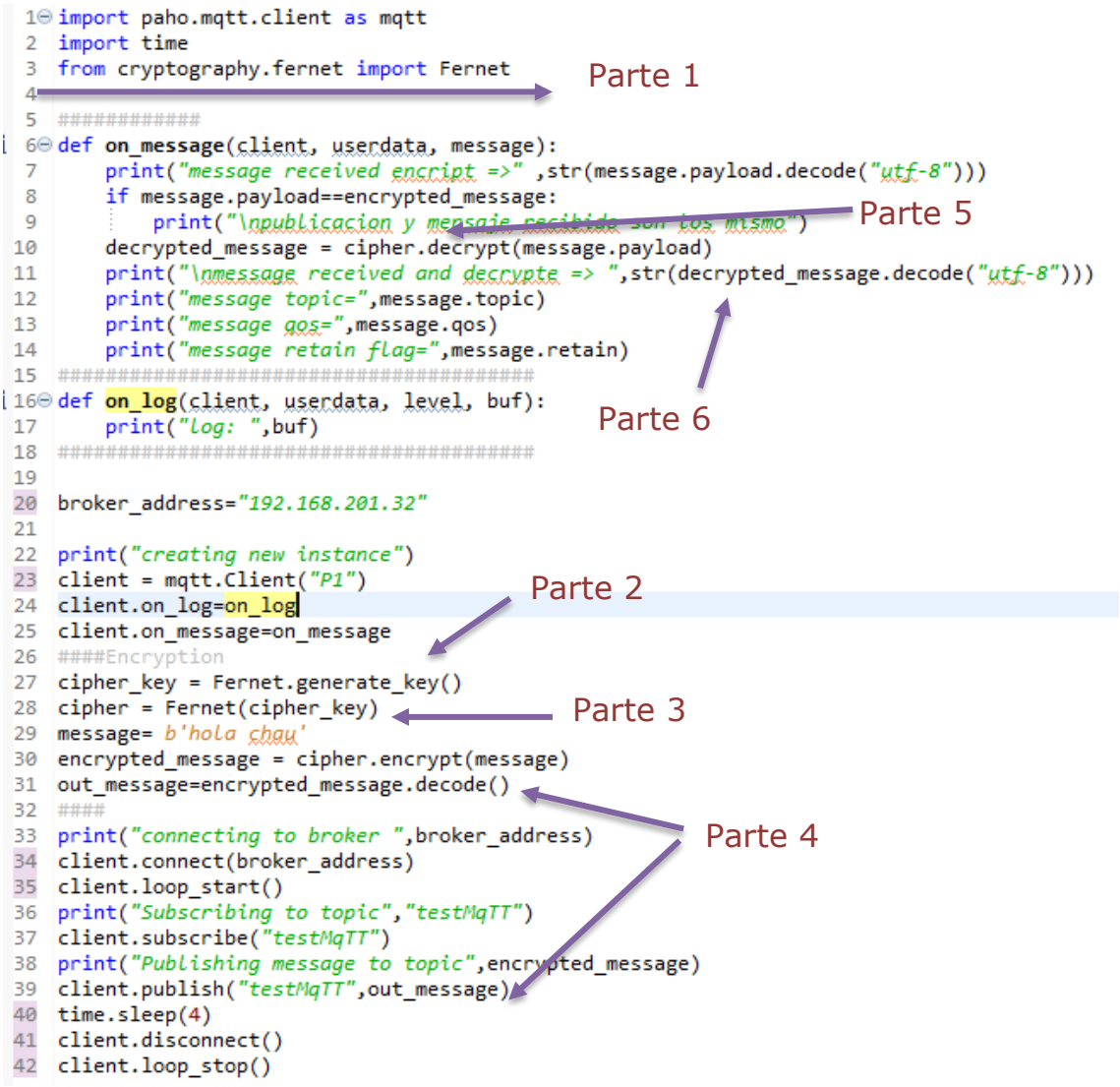
En efecto, tiene la ventaja de que los datos se cifran de extremo a extremo y no solo entre el agente y el cliente sino también significa que los intermediarios no necesitan admitir certificado SSL permitiendo evitar obtener, instalar y comprar los certificados.

Por esa razón, es mucho más fácil de configurar y usar sin complicaciones, protegiendo la información recopilada por los sensores y luego enviadas a través del protocolo MqTT.

A continuación se puede observar parte de código de un script con librería de Python para poder realizar el cifrado simétrico con encriptación AES. También se realiza una breve explicación de parte del contenido del código utilizado. Con ésta técnica los datos específicos de la aplicación se cifran a nivel de la aplicación cifrando la carga útil del

mensaje. En la figura 4 se puede ver el script del código y luego una breve explicación del mismo.

Figura 4: Scrip del código.




```
import paho.mqtt.client as mqtt
import time
from cryptography.fernet import Fernet

############
def on_message(client, userdata, message):
    print("message received encript =>" ,str(message.payload.decode("utf-8")))
    if message.payload==encrypted_message:
        print("\npublicacion y mensaje recibido son los mismo")
    decrypted_message = cipher.decrypt(message.payload)
    print("\nmessage received and decrypte => ",str(decrypted_message.decode("utf-8")))
    print("message topic=",message.topic)
    print("message qos=",message.qos)
    print("message retain flag=",message.retain)
########################################
def on_log(client, userdata, level, buf):
    print("log: ",buf)
########################################

broker_address="192.168.201.32"

print("creating new instance")
client = mqtt.Client("P1")
client.on_log=on_log
client.on_message=on_message
####Encryption
cipher_key = Fernet.generate_key()
cipher = Fernet(cipher_key)
message= b'hola chau'
encrypted_message = cipher.encrypt(message)
out_message=encrypted_message.decode()
####
print("connecting to broker ",broker_address)
client.connect(broker_address)
client.loop_start()
print("Subscribing to topic","testMqTT")
client.subscribe("testMqTT")
print("Publishing message to topic",encrypted_message)
client.publish("testMqTT",out_message)
time.sleep(4)
client.disconnect()
client.loop_stop()
```

Fuente: Elaboración propia.

- Parte 1: Se ha utilizado el paquete de criptografía para realizar el cifrado.

- Parte 2: Primero se crea una clave de cifrado: cipher_key = Fernet.generate_key () . Esta clave se usa para cifrar y descifrar y se necesitaría usar esta misma clave en el cliente receptor.

- Parte 3: el mensaje a cifrar debe estar en bytes.

- Parte 4: Es necesario crear una cadena codificada UTF-8 para pasarla como carga del mensaje al método de publicación MQTT.

- Parte 5: el mensaje recibido ya está en bytes, por lo que lo ahora pasa directamente a la función de descifrado.

- Parte 6: Luego se convierte el mensaje de byte descifrado a una cadena UTF-8 de manera normal

Luego, al ejecutar el script se crea una nueva instancia. En la figura 5 se puede ver esta como queda luego de esta acción.

Figura 5: Scrip luego de su ejecución.

```
creating new instance
connecting to broker  192.168.0.201
log:  Sending CONNECT (u0, p0, wr0, wq0, wf0, c1, k60) client_id=b'P1'
Subscribing to topiclog:  testMqTT
log:  Sending SUBSCRIBE (d0, m1) [(b'testMqTT', 0)]
 Received CONNACK (0, 0)
Publishing message to topiclog:   b'gAAAAABddGqMJu0t92bw3o19DKhaTtCvDpn6Vr-aNwZL6IzfxC9FOb-gfPx66L2eWyLx3nr1fWXqo1Cg1KR5HWKQmWmbu837OQ=='
log:  Sending PUBLISH (d0, q0, r0, m2), 'b'testMqTT'', ... (100 bytes)
Received SUBACK
log:  Received PUBLISH (d0, q0, r0, m0), 'testMqTT', ...  (100 bytes)
message received encript => gAAAAABddGqMJu0t92bw3o19DKhaTtCvDpn6Vr-aNwZL6IzfxC9FOb-gfPx66L2eWyLx3nr1fWXqo1Cg1KR5HWKQmWmbu837OQ==

publicacion y mensaje recibido son los mismo

message received and decrypte =>  hola chau
message topic= testMqTT
message qos= 0
message retain flag= 0
log:  Sending DISCONNECT
```

Fuente: Elaboración propia.

Finalmente en las figuras 6, y 7 se puede observar distintas vistas del mensaje cifrado con encriptación AES.

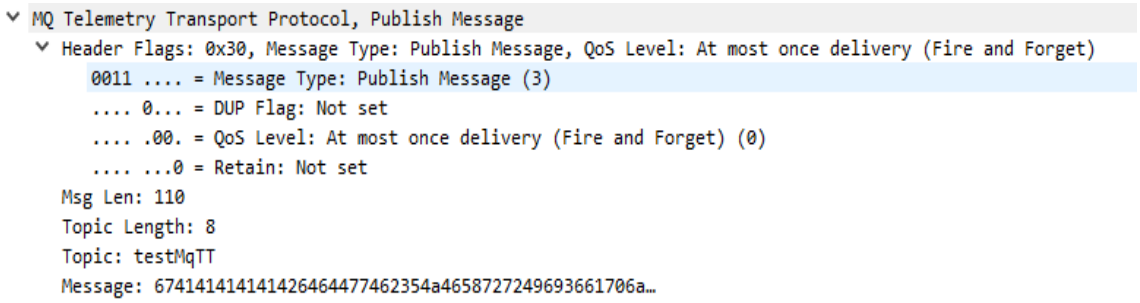
Figura 6: Mensaje cifrado

```
C:\Program Files\mosquitto>mosquitto_sub.exe -t "testMqTT" -v
testMqTT ko
testMqTT gAAAAABddGqMJu0t92bw3o19DKhaTtCvDpn6Vr-aNwZL6IzfxC9FOb-gfPx66L2eWyLx3nr1fWXqo1Cg1KR5HWKQmWmbu837OQ==
```

Fuente: Elaboración propia.

En la figura 7 se puede observar otra vista del mensaje cifrado.

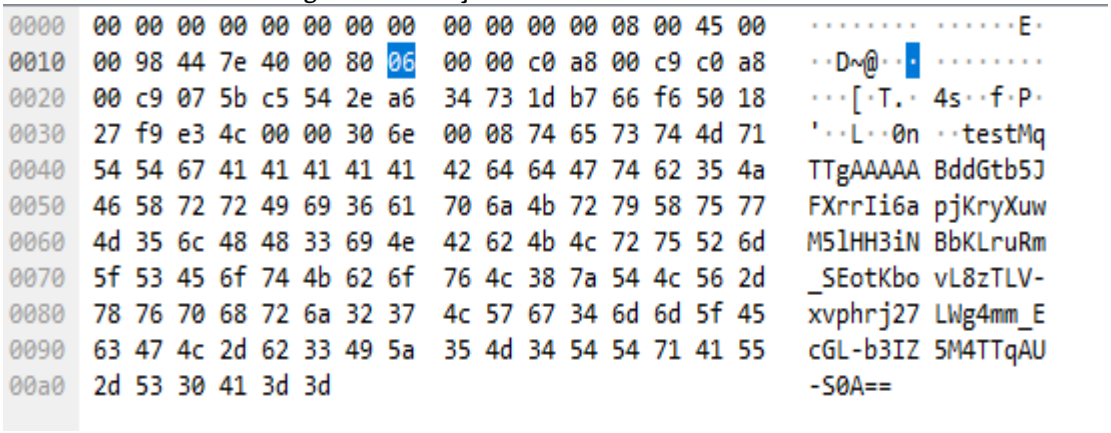
Figura 7: Otra vista del mensaje cifrado.



Fuente: Elaboración propia.

Finalmente en la figura 8 se puede observar el mismo mensaje cifrado, visto desde la aplicación Wireshark.

Figura 8: Mensaje cifrado visto desde Wireshark.



Fuente: Elaboración propia.

De esta manera queda demostrada la vulnerabilidad del protocolo Mqtt y una posible solución de asegurar el mensaje con una capa de seguridad empleando técnica de cifrado simétrico con encriptación AES.

CONCLUSIONES

El presente trabajo se centra en mostrar otra posible solución, a las ya existentes, en lo referente a la mejora en la seguridad de las comunicaciones en Internet de las Cosas. Para ello y mediante la manipulación de un robot, se hizo uso del protocolo Mqtt para envío de mensajes habiéndose demostrado la vulnerabilidad existente en el mismo, al quedar expuesto los mensajes cuando se realizó una intercepción de los mismos.

De la misma manera también se realizó la prueba de la solución propuesta mediante la aplicación de una capa de seguridad en los mensajes con aplicación de técnicas de cifrado simétrico con encriptación AES para lo cual se consiguió resultado positivo ya que el mensaje no quedó expuesto a la lectura ante una eventual intercepción al mismo.

Si bien, en los últimos tiempos se ha logrado avances significativos en lo referente a seguridad en los sistemas, aún queda mucho por investigar y desarrollar nuevas técnicas que ayuden a mejor la protección tanto de datos como de dispositivos en el área de Internet de las Cosas para que de esa manera poder acercarnos mas a los principios fundamentales de la seguridad que es proporcionar confidencialidad, integridad y disponibilidad de los recursos críticos.

BILBIOGRAFIA

[1] Chen, Xu, Liu, Hu, & Wang. (2014). A vision of IoT: Applications, Challenges, and Opportunities With China Perspective. IEEE Internet of Journal, 349 - 359.

[2] Alvear Puertas, R. M. (2017). Internet de las Cosas y Visión Artificial, Funcionamiento y Aplicaciones: Revisión de Literatura. Enfoque UTE, 244 - 256.

[3] Jaehak Byun, S. K. (2016). Models, Smart City Implementation. Mechanical Engineering, 209 -212.

[4] Karen Rose, S. E. (2015). La Internet de las cosas. Internet Society, 210 - 215.

[5] Hyewon Jeong, J. L.-C. (2017). A Low-Power High-Performance SoC Platform. IDEC Journal of Integrated Circuits and Systems,, 21 - 23.

[6] Franco, R. (2015). Internet de las cosas. Universidad Catolica Nuestra Señora de la Asunción, 11 - 12.

[7] Perez, N. B. (2018). Análisis sistemático de la seguridad en internet of things.

[8] Eterovic, J. C. (2019). Análisis de la seguridad de los datos en Internet de las Cosas usando tecnología blockchain.