UNIOESTE

Universidade Estadual
do Oeste do Paraná

IoT: protocolo MQTT

Professor:
Renato Bobsin Machado

Alunos:
Larissa L. Wong,
Marco A. Guerra,
Victor E. Almeida

April 30, 2022

Conteúdo

Lista de Figuras

Lista de Tabelas

1 Introdução

O termo *Internet of Things (IoT)*, do português *Internet das Coisas* foi proposto pelo professor britânico, Kevin Ashton do Instituto de Tecnologias de Massachusetts (MIT), no ano de 1999[1] e diz a respeito da forma geral de uma rede que conecta diversos dispositivos, "coisas", à internet, através de software, com o objetivo de realizar troca informações[2] gerando valor e conhecimento. Dessa forma, um mesmo sistema passava a ser composto por um rede dispersa de dispositivos, como podem ser: sensores, microcontroladores ou eletrodomésticos - como geladeiras, televisores, entre outros.

Camadas dos sistemas envolvendo IoT

A estrutura geral dos sistemas envolvendo a *Internet das Coisas*, como proposto por Ashton pode ser dividida em três camadas principais[1] que caraterizam as diferentes responsabilidades do sistema:

- Em primeiro lugar, a **Camada de Percepção**, responsável pela captação dos dados do meio, envolve os sensores e o hardware necessários para a obtém-são da informações relevantes a respeito de fenômenos dispersos e de diferentes naturezas. Exemplos de fenômenos de interesse para os sistemas de IoT são os eventos meteorológicos, biológicos ou físicos como, a temperatura e umidade do solo[3], do ar[4], índice de área foliar[5], quantidade de gás SO_2[6], PH do solo[6] entre muitos outros, dependendo do tipo de aplicação que tem por objetivo o sistema.

- Em segundo lugar, a **Camada de Comunicação**, é a responsável pela transmissão dos dados oriundos dos diversos dispositivos conectados com outras camadas, como são: os sensores da camada de percepção, os servidores responsáveis por fazer a análise dos dados, as aplicações encarregadas de disponibilizar a inteligência obtida a partir dos dados, ou ainda outros dispositivos que atuarão em dependência dos valores recebidos. Nesse sentido, é na junção da Camada de Comunicação e a Camada de Percepção que acabam por caraterizar um sistema como sendo de IoT.

- Em terceiro lugar, e finalmente, a **Camada de Aplicação** é a encarregada de dar sentido aos dados coletados pelos sensores, pois é nesse momento que ocorre a apresentação dos mesmos de forma a gerar valor para os usuários do sistema. Dessa forma, uma área onde tem ocorrido uma ampla implementação dos sistemas IoT é na agricultora. Tal

fato fica justificado pela relevância e singularidade dos dados que caraterizam uma plantação, os quais variam rapidamente no tempo e no espaço. Assim, a partir dessa informação pode ser identificado qual é o melhor momento e lugar da fazenda realizar o plantio e colheita das diferentes culturas[7], ou mesmo em quais lugares da plantação ha doenças[8].

No contexto dos sistemas de Internet das Coisas, a camada de comunicação é a que maiores desafios oferece para os desenvolvedores de soluções desse tipo de sistemas. Isso se justifica, tendo em vista os requisitos como são: as longas distâncias de transmissão de dados, o uso de dispositivos de baixo custo, busca por baixos consumos de energia, entre outros. Nesse sentido, o protocolo de comunicação MQTT, originalmente conhecido como *MQ Telemetry Transport*, tem se tornado praticamente um estândar entre as soluções de IoT, desde que foi proposto.

2 História do protocolo MQTT

O protocolo MQTT começou a ser idealizado e planejado durante a década de 1990[9], pelos engenheiros Andy Stanford-Clark da IBM e Arlen Nipper da Cirrus Link/Eurotech[10].

Andy e Arlen se estavam trabalhando na conexão de oleodutos via satélite, os quais visto a limitação das conexões e o hardware da época perceberam a necessidade de um protocolo de comunicação que utilizasse pouca largura de banda e pouca energia do dispositivo, tendo isso em mente elicitaram os seguintes requisitos[10]:

- Implementação simples;
- Uso de QoS, *Quality of Service* por quem publica a mensagem;
- Uso eficiente de largura de banda, baixo *overhead*;
- Baixo custo energético para envio;
- Possibilidade de enviar qualquer tipo de dado;
- Possibilidade de manter conexões ativas, prontas para enviar e receber dados;

Uma vez projetado, foi implementado sua primeira versão no ano de 1999 e batizado MQTT, *MQ Telemetry Transport*, em referência ao produto da IBM MQ Series, que atua na camada de transporte sendo uma ferramenta importante em arquiteturas orientadas a serviços[11].

Esse protocolo de comunicação foi muito utilizado dentro da IBM em diversas aplicações, produtos e serviços desde sua criação até 2010[10]. Porém nesse momento da história viu-se a possível utilidade da tecnologia em aplicações IoT, com isso o MQTT deixou de ser *software* proprietário apenas utilizado em sistemas embarcados da IBM, para ser um dos principais protocolos utilizados em internet das coisas, com isso foi lançado o MQTT 3.1 o qual pode ser utilizado sem precisar pagar por *royaltys* ou taxas de propriedade.

Com a abertura do protocolo a IBM fez investimentos para criar um ecossistema colaborativo para incentivar o seu uso, principalmente através da *Eclipse Foundation*, uma corporação sem fins lucrativos que visa impulsionar projetos de software livre e tecnologias abertas[12], que gerência mais de 400 repositórios abertos[1], muitos deles relacionados ao MQTT, como por exemplo:

[1]Github da Eclipse Foundation: <https://github.com/eclipse>

- **Bibliotecas Paho**[2]: Tendo inicio em 2012/2013 o projeto Paho disponibiliza um conjunto de bibliotecas que implementam clientes MQTT em diversas linguagens de programação, sendo a mais conhecida delas em Python, porém foram escritas versões para C++, C, Rust, Java, Go, JavaScript, Ruby, D...

- **Broker Mosquitto**[3]: Servidor que implementa o protocolo MQTT, também chamado de Broker. Por ser *open source* possuí diversos *forks* e customizações, bem como bibliotecas para facilitar seu uso e configuração.

- **Visualizador de dados Streamsheet**[4]: Aplicação JavaScript que permite obter os dados vindos de dispositivos que se comunicam com MQTT, uma plataforma na qual não é necessário escrever código, basta realizar ações de maneira gráfica.

Além da implementação dos projetos supracitados a IBM também trabalhou por padronização no protocolo, em 2013 foi anunciado que o MQTT seria padronizado pela OASIS[10], empresa sem fins lucrativos que visa padronizar projetos abertos com aprovação de regras de políticas internacionais[13].

Após mais de um ano no dia 29 de outubro de 2014 o MQTT foi aprovado como padrão pela OASIS na sua versão 3.1.1, onde foram adicionados novos requisitos como clientes anônimos, mais códigos de erro ao se inscrever em determinado tópico entre outros.

De 2014 até hoje o MQTT passou por diversas atualizações e melhorias, sendo 5.0 a última versão ratificada pela OASIS em março de 2019[10], especificando requisitos necessários para dispositivos IoT modernos, como tratamento de erros e interação com plataformas em nuvem[10].

evolução histórica da tecnologia

[2]Implementação da biblioteca em Python:<https://github.com/eclipse/paho.mqtt.python>

[3]Código fonte do Broker: <https://github.com/eclipse/mosquitto>

[4]Github do projeto: <https://github.com/eclipse/streamsheets>

3 Aplicações

O protocolo MQTT é utilizado em diversas aplicações IoT em várias escalas incluindo a escala industrial. Entre os usos industriais há grandes projetos nas áreas automotivas, logísticas, indústrias, monitoramentos com Smart Home, petrolíferas e aplicações eletrodomésticas. [14]

São exemplos de aplicações consolidadas no mercado: Serviço de compartilhamento de carros da BMW (BMW Mobility Services) e Transporte de amostras entre hospitais e laboratórios utilizando drones autônomos da Matternet. [14]

Ambas as aplicações escolhidas utilizam o protocolo MQTT por meio de uma plataforma chamada HiveMQ, essa plataforma fornece ao usuário um broker MQTT e uma plataforma de mensagem para facilitar o envio e recebimento dos dados de dispositivos IoT, fornecendo dashboards como forma de monitoramento dos dados que passam entre o broker e os clientes conectados com a aplicação. [14, 15]

3.1 BMW Mobility Services

Figura 1: BMW Mobility Services.

A BMW Mobility Services é um grupo empresarial interno ao grupo BMW que desenvolve soluções de mobilidade para população urbana. Entre as soluções de mobilidade desenvolvidas está o serviço de compartilhamento de veículos para operadores de frota. Com esse serviço os operadores de frota podem realizar diversas tarefas de maneira remota, como abrir ou fechar os veículos e obter dados de navegação de cada um. [16]

Utilizando esse desenvolvimento o grupo BMW lançou um serviço chamado DriveNow que está presente em 12 cidades na Europa. [16]

O uso da HiveMQ foi incluído na plataforma de mensagens do compartilhamento de veículos em 2014. Em razão do sucesso do uso do protocolo MQTT e da plataforma foi realizada a extensão do uso de ambos para mensagens internas que enviam dados entre microsserviços de backend da BMW Mobility Services. [16]

3.2 Matternet Autonomous Drones

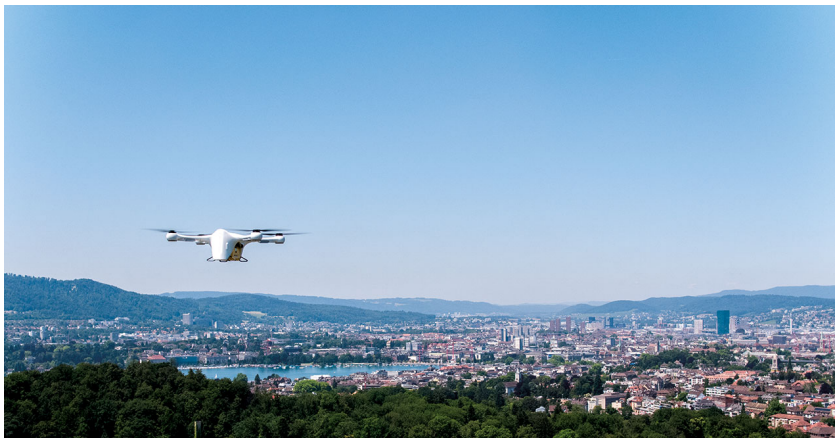

Figura 2: Matternet Autonomous Drones.

A Matternet é uma startup, localizada na Califórnia, que utiliza drones autônomos para transporte de amostras entre hospitais e laboratórios. Essa forma de transporte se mostra consideravelmente mais eficiente que as formas convencionais (carros, ônibus ou motos) pois o tempo relativo à entrega é reduzido em razão da eliminação de fatores como congestionamentos e desvios na rota. Com os drones é possível realizar voos que seguem uma rota direta para o destino, diminuindo assim a distância percorrida. [17]

Em razão dos drones realizarem os voos de forma autônoma não é necessário que um piloto realize as funções, porém órgãos reguladores locais de Zurich exigem que um operador humano tenha acesso em tempo real a dados do drone como posição, além de conseguir realizar o controle pilotando-o se for necessário em intercorrências. Além da preocupação governamental relacionada ao modo de voo também é necessário que os voos sejam sincronizados com o espaço aéreo de outras aeronaves, pois os drones podem realizar voos em regiões de hospitais em que há helipontos. [17]

Para que o projeto fosse executado foi necessária a montagem de um esquema com drones autônomos, estações de pouso e uma plataforma logística

baseada em nuvem. Essa plataforma logística utiliza o protocolo MQTT juntamente com HiveMQ para comunicação em tempo real dos voos. Os drones enviam mensagens MQTT para o broker da HiveMQ da plataforma logística, como o protocolo suporta comunicação bidirecional é possível realizar a leitura dos dados e controle dos drones caso necessário juntamente com controle do tráfico aéreo. [17]

4 Embasamento teórico

4.1 Terminologia

Para um melhor entendimento do protocolo de comunicação MQTT é necessário entender a terminologia própria utilizada pelo protocolo. Dessa forma, alguns dos termos de interesse são:

- **Conexão de rede**: se refere ao serviço provido pelo protocolo de comunicação na camada subjacente (TCP/IP) ao protocolo MQTT, e que permite o envio de informação entre os dispositivos.
- **Mensagem**: representa a informação que se deseja transmitir e que portanto constitui o objetivo da comunicação. De forma geral, e pelas caraterísticas do protocolo, a mensagem deve ocupar o mínimo espaço possível - na maioria dos casos é um único valor.
- **Cliente**: programa ou dispositivo que irá enviar ou receber informação. Nesse sentido o cliente deve ser capaz de:
  - iniciar uma conexão com o servidor,
  - realizar a publicação de informações em tópicos,
  - realizar a subscrição a tópicos de interesse,
  - realizar o cancelamento de uma subscrição e,
  - finalizar uma conexão com o servidor.
- **Broker**: programa ou serviço que intermediá e gerência a informação, o envio e recebimento de dados dos clientes, é comum se referir ao mesmo como servidor. Nesse sentido o servidor deve ser capaz de:
  - gerencia as conexões como os clientes,
  - gerenciar as mensagens publicadas pelos clientes,
  - gerenciar as subscrições dos clientes e,
  - retransmitir as mensagens recebidas aos clientes.
- **Sessão**: se corresponde com o período de interação entre um cliente e um servidos, durante o qual é mantido um grupo de informações que representam o estado da comunicação, podendo ser composto por mais de uma conexão.

- **Subscrição**: é o ato de que o cliente desempenha para indicar ao servidor interesse em receber atualizações sobre a mudança de um tópico. Dessa forma, uma subscrição se encontra composta por um Filtro de Tópicos, para um ou mais tópicos, e um valor indicando a Qualidade de Serviço deseja (QoS).

- **Subscrição compartilhadas**: são subscrições que se encontram associadas com mais de uma sessão de comunicação entre o cliente e o servidor. Dessa forma, permitem um maior rango de padrões de comunicação.

- **Caracteres mágicos**: também chamados de Wildcards, permitem ao cliente indicar interesse em receber notificações de mais um tópico, para isso fazendo uso de caracteres que definem o padrão dos tópicos desejados.

- **Nome de Tópico**: rótulo ou identificador de uma informação específica dentro do broker e que é constantemente atualizada pelos cliente quando publicam uma nova informação no tópico.

- **Filtro de Tópico**: é uma expressão contida numa subscrição é que se corresponde com um ou mais tópicos

- **Paquete MQTT**: conjunto de informação útil à comunicação cliente-servidor envida através da conexão de rede. O protocolo MQTT especifica 15 tipos de pacotes diferentes, os quais serão especificados a continuação.

5 Características técnicas

5.1 Formato dos pacotes MQTT

O envio de dados através do protocolo MQTT utiliza diferentes tipos de pacotes, cada um deles contendo obrigatoriamente um cabeçalho de tamanho fixo, o qual pode ser complementado por um segundo cabeçalho de tamanho variável e um bloco de carga útil, ambos opcionais.

Tabela 1: Estrutura básica comum a todos os pacotes MQTT.

Cabeçalho fixo
Cabeçalho variável
Carga útil

Cabeçalho fixo

Dentro do protocolo MQTT é no cabeçalho fixo onde é especificado o tipo de pacote que esta sendo enviado, e portanto a sua função no processo de comunicação.

Tabela 2: Estrutura do cabeçalho fixo utilizado no protocolo MQTT.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Byte 1	**Tipo de pacote MQTT**				**Sinais específicos do pacote**			
Byte 2	**Espaço restante**							
. . .								

Tipo de pacote MQTT

Se encontra especificado nos 4 bits mais significativos do primeiro byte da cabeçalho fixo, os quais codificam um número inteiro entre 0 e 15, que por sua vez representa o tipo de pacote, como relacionado na tabela a continuação.

Tabela 3: Tipos de pacotes utilizados pelo protocolo MQTT.

Tipo	Código	Remetente	Descrição
Reserved	0	-	Reservado
CONNECT	1	Cliente	Solicitação de conexão
CONNACK	2	Servidor	Confirmação de conexão
PUBLISH	3	Ambos	Publicar mensagem
PUBACK	4	Ambos	Publicar confirmação (QoS 1)
PUBREC	5	Ambos	Publicar recebimento (QoS 2 – parte 1)
PUBREL	6	Ambos	Publicar lançamento (QoS 2 – parte 2)
PUBCOMP	7	Ambos	Publicar conclusão (QoS 2 – parte 3)
SUBSCRIBE	8	Cliente	Solicitação de inscrição
SUBACK	9	Servidor	Confirmação de inscrição
UNSUBSCRIBE	10	Cliente	Solicitação de cancelamento de inscrição
UNSUBACK	11	Servidor	Confirmação de cancelamento de inscrição
PINGREQ	12	Cliente	Solicitação de PING
PINGRESP	13	Servidor	Resposta de PING
DISCONNECT	14	Ambos	Notificação de desconexão
AUTH	15	Ambos	Troca de autenticação

Sinais específicos do pacote

Complementa a informação do tipo do pacote, e se encontra representado pelos 4 bits menos significativos do primeiro byte. Dessa forma, indica diferentes modos em que pode ser usado o mesmo formato de pacote.

Espaço restante

Informação útil e de controle dependentes de cada tipo de pacote.

5.2 Função dos pacotes MQTT

- **CONNECT (Requisição de Conexão)**: é o primeiro pacote que deve ser enviado pelo cliente no momento após ter estabelecido um conexão de rede com servidor. Entre as informações enviadas se encontra um identificador único para o cliente, entre outras informações opcionais como usuário e senha.

- **CONNACK (Reconhecimento de conexão)**: é o pacote enviado pelo servidor em resposta ao recebimento do pacote CONNECT enviado pelo cliente, e que deve ser recebido antes do pacote AUTH. Em caso do cliente não receber um pacote do tipo CONNACK em um tempo razoável o mesmo deve encerar a conexão com o servidor e tentar novamente.

- **PUBLISH (Publicar mensagem)**: são os pacotes encarregados de carregar as mensagens entre o servidor e os clientes podendo ir em ambas as direções.

- **PUBACK (Reconhecimento de publicação)**: é o pacote enviado em resposta a um PUBLISH realizado com QoS 1.

- **PUBREC (Recebimento de publicação)**: é o pacote enviado em resposta a um PUBLISH representa a segunda parte do intercambio de mensagens realizado com QoS 2.

- **PUBREL (Liberação de publicação)**: é o pacote enviado em resposta a um PUBREC e representa a terceira parte do intercambio de mensagens realizado com QoS 2.

- **PUBCOMP (Publicação completada)**: é o pacote enviado em resposta a um PUBREL e representa a quarta e última parte do intercambio de mensagens realizado com QoS 2.

- **SUBSCRIBE (Requisição de subscrição)**: é o pacote enviado pelo cliente ao servidor como solicitude para a criação de uma ou mais assinaturas, registrando o interesse em receber informação de um ou mais tópicos, junto com o tipo de qualidade do serviço máxima (QoS) desejada com a qual o servidor pode enviar uma mensagem ao cliente.

- **SUBACK (Reconhecimento de subscrição)**: é o pacote enviado pelo servidor ao cliente com o objetivo de confirmar o recebimento e

processamento de um pacote do tipo SUBSCRIBE. Entre as informações contidas pelo pacote tem uma lista indicando a qualidade do serviço concedida ao cliente para cada tópico e em caso de ter sido negada a subscrição é enviado um sinal justificando o motivo do mesmo.

- **UNSUBSCRIBE (Requisição de cancelamento de subscrição)**: é o pacote utilizado pelo cliente para indicar que já não deseja receber atualizações da mudança de valore de um tópico.
- **UNSUBACK (Reconhecimento de cancelamento de subscrição)**: é o pacote enviado pelo servidor ao cliente indicando conhecimento do pedido de anulação de subscrição iniciado pelo pacote de UNSUBSCRIBE.
- **PINGREQ (Requisição de PING)**: é o pacote envido pelo cliente ao servidor, é pode ser utilizado para: indicar ao servido que o cliente se encontra ativo e à espera de qualquer pacote MQTT, requerer ao servidor que responda se se encontra ativo, ou, ainda, para verificar se a conexão à rede se encontra em funcionamento.
- **PINGRESP (Resposta a uma Requização de PING)**: é a resposta enviada pelo servidor quando recebe uma mensagem do tipo PINGREQ, sinalizando que o mesmo se encontra em funcionamento.
- **DISCONNECT (Requisição de desconexão)**: é o pacote de controle final enviado pelo cliente ao servidor, indicando a intenção e o motivo pelo qual a conexão de rede está sendo fechada.
- **AUTH (Intercambio de autentificações)**: é pacote enviado para a troco de informação referente a autentificação de informação, servindo dessa forma para a prevenção de erros de comunicação.

6 Segurança

O desenvolvimento rápido de dispositivos IoT e o grande crescimento desse ramo da tecnologia no mercado se deu muitas vezes em detrimento da consideração de segurança e limitações dos dispositivos. Esse aspecto, juntamente com o fato desse alto número de dispositivos se conectar à Internet e ter capacidade de armazenar diversas informações dos usuários traz grande risco a ataques. [18]

Como exemplo de ataque à dispositivos inteligentes há o ataque à empresa Dyn DNS. A empresa sofreu um ataque de negação de serviço distribuído (Distributed Denial of Service – DDoS) orquestrado por uma botnet. Nesse ataque o malware atingiu diversos dispositivos por meio de um ataque de força bruta em que realizava o acesso à interface do dispositivo e realizava upload de seu código. Esse DDoS conseguiu gerar um fluxo de pacotes TCP e UDP e atacar uma porta com sucesso. Esse ataque e vários outros demonstram a importância da segurança em dispositivos IoT. [18, 19]

Os DDoS estão entre os tipos mais comuns de ataques na Internet em que há uma tentativa maliciosa de interromper o tráfego de servidores, serviços ou rede por meio da sobrecarga no destino com enxurradas de tráfego. A botnet é um grupo de dispositivos infectados com malware que passa a realizar o ataque, quando é estabelecida a botnet o invasor direciona instruções para cada dispositivo infectado. Assim um endereço de IP pode receber solicitações de diversos dispositivos controlados pelo malware ao ponto de extrapolar a capacidade do servidor. Como cada robô (bot) é um dispositivo de Internet legítimo não é uma tarefa fácil para o servidor ou rede de destino separar os dispositivos que realizam os ataques de dispositivos inocentes. [19, 20]

O protocolo MQTT possui suporte para diferentes mecanismos de segurança, porém a maior parte deles não é configurada como padrão como criptografia de dados e autenticação de entidades. Cada cliente MQTT possui um ID único de cliente e, se houver tentativa de conexão com um ID já conectado, a primeira conexão é encerrada. Os mecanismos de autenticação que usam o endereço de controle de acesso ao meio (MAC – Media Acess Control) são controlados pelos brokers que registram essa informação dos dispositivos quando há tentativa de conexão. A autorização de acesso pode ser realizada utilizando mecanismos de lista de controle de acesso (ACL – Acess Control Lists), que promove um meio de proteção de informações em que é possível restringir o acesso a diferentes porções de diretórios ou entradas para estes diretórios. [18, 21]

A confiabilidade também é um aspecto importante para sistemas seguros

e pode ser tratada na camada de aplicação com a realização da criptografia da mensagem a ser publicada. Essa criptografia pode ser implementada ponta a ponta ou cliente a broker. Na criptografia cliente-broker o broker realiza a descriptografia da informação que está sendo publicada e criptografa valores que serão enviados a outros clientes. Na cripotrafia ponto a ponto o broker não descriptografa a informação, apenas enviando o texto criptografado a outros dispositivos, utilizando menos recursos computacionais e menos energia por trabalhar apenas como mensageiro. [18]

Quanto a camadas mais baixas também há possibilidade de adição de mecanismos de segurança. Uma forma confiável de garantir a segurança de canais de comunicação na camada de transporte é utilizar o protocolo Transport Layer Security (TLS) para Transmission Control Protocol (TCP), que permite que duas partes identifiquem e autentiquem uma a outra por meio de certificados digitais e realizem comunicação com confidencialidade e integridade de dados. No caso do uso de User Datagram Protocol (UDP) uma forma de garantir a segurança é o uso de Datagram Transport Layer Security (DTLS) que é baseado no protocolo TLS e fornece garantias de segurança da mesma forma que o TLS. [22, 23, 24]

Um desafio relacionado à implantação de bons mecanismos de segurança é a falta de recursos pois, tomando como exemplo a implementação do protocolo TLS em que o overhead gerado pode ser muito grande para dipositivos pequenos e com recursos limitados. Outros dispositivos IoT podem nem oferecer suporte a esse protocolo. [18]

Podem ser utilizados diversos algoritmos como Advanced Encryption Standard (AES) como algoritmos de criptografia, sendo o AES o mais utilizado. [25]

Dentre as formas de aumentar a segurança com o protocolo MQTT estão: autenticação e autorização de clientes pelo servidor, autenticação do servidor pelos clientes, verificação da integridade dos dados por meio de valores de hash, uso de VPN, realizar a detecção de comportamentos anormais como muitas tentativas de conexão, entre outros. [25]

Outra forma de aumentar a segurança é o uso do protocolo Secure MQTT, nas quais a implementação do servidor oferece TLS e utilizam a porta 8833. O uso nesse formato possui cadastro na Autoridade para Atribuição de Números da Internet (IANA – Internet Assigned Numbers Authority) como secure-mqtt. A IANA supervisiona a atribuição global dos números na Internet como endereços IP, servidores raiz de DNS e registros de protocolos de comunicação como no caso do MQTT. [25, 26]

7 Exemplo Prático

7.1 Descrição da aplicação implementada

A modo de exemplo é apresentada uma aplicação que a partir de um ESP32 realiza a leitura da temperatura do ambiente fazendo uso de um sensor BMP280 e envia ela como um tópico para um broker que se encontra conectado na mesma rede e que por sua vez tem a capacidade de alterar o estado de um LED conectado ao ESP32.

7.2 Descrição das tecnologias e dispositivos utilizados

Para implementar a aplicação apresentada foram utilizados os seguintes componentes e tecnologias:

- Esp32:
  - Sensor Adafruit BMP-280;
  - Led embutido no Esp32;
  - C/C++ (Framework Arduino e ESP-IDF);
- Raspberry:
  - Docker executando o broker mosquitto;
  - Cliente inscrito no tópico "#"

7.3 Esquema da aplicação utilizada

Dessa forma, o funcionamento da aplicação implementada se encontra resumido no seguinte esquema:

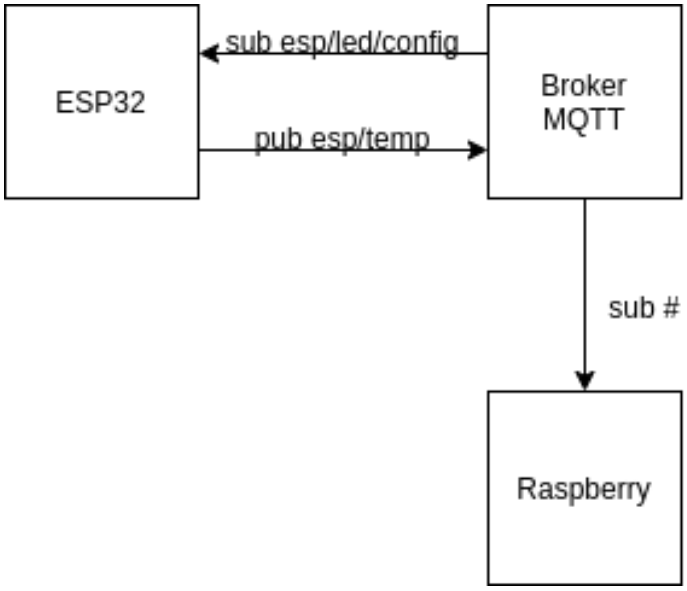

Figura 3: Dispositivos e tópicos utilizados

7.4 Métodos principais da aplicação

```
1  void setup() {
2      if (!sensor.begin(BMP280_ADDRESS)) {
3          if(!sensor.begin(BMP280_ADDRESS_ALT)) {
4              delay(1000);
5              ESP.restart();
6          }
7      }
8      pinMode(LED_PIN, OUTPUT);
9      wifiConnect();
10     MqttConnect();
11     xTaskCreate(taskSendTemperature, "send", 20000, NULL, 1,
           &handle);
12 }
13
14 void loop() {
15     if (!mqttClient.connected()) {
16         MqttConnect();
17     }
18     mqttClient.loop();
19 }
```

8 Conclusões

O presente trabalho foi desenvolvido com o intuito de ajudar a esclarecer e apresentar aspectos gerais e principalmente técnicos a respeito do protocolo de comunicação MQTT. Dessa forma, serviu para trazer um melhor entendimento sobre o mesmo, ajudando a esclarecer aspectos como a sua história, caraterísticas principais, contexto de uso, tipos de pacotes utilizados, entre outros. Por outro lado, o presente trabalho foi útil para o desenvolvimento de capacidades necessárias para o trabalho em equipe por partes dos participantes.

Referências

1 TZOUNIS, A. et al. Internet of things in agriculture, recent advances and future challenges. **Biosystems Engineering**, v. 164, p. 31 – 48, 2017. ISSN 1537-5110. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1537511017302544>. Acesso em: 3 de dezembro de 2020. Citado na página 4.

2 Chen, S. et al. A vision of iot: Applications, challenges, and opportunities with china perspective. **IEEE Internet of Things Journal**, v. 1, n. 4, p. 349–359, 2014. Disponível em: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6851114>. Acesso em: 16 de janeiro de 2021. Citado na página 4.

3 KIZITO, F. et al. Frequency, electrical conductivity and temperature analysis of a low-cost capacitance soil moisture sensor. **Journal of Hydrology**, v. 352, n. 3, p. 367 – 378, 2008. ISSN 0022-1694. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0022169408000462>. Acesso em: 3 de dezembro de 2020. Citado na página 4.

4 MESAS-CARRASCOSA, F. et al. Open source hardware to monitor environmental parameters in precision agriculture. **Biosystems Engineering**, v. 137, p. 73 – 83, 2015. ISSN 1537-5110. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1537511015001208>. Acesso em: 3 de dezembro de 2020. Citado na página 4.

5 BAUER, J. et al. On the potential of wireless sensor networks for the in-situ assessment of crop leaf area index. **Computers and Electronics in Agriculture**, v. 128, p. 149 – 159, 2016. ISSN 0168-1699. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0168169916306585>. Acesso em: 3 de dezembro de 2020. Citado na página 4.

6 KARIMI, N. et al. Web-based monitoring system using wireless sensor networks for traditional vineyards and grape drying buildings. **Computers and Electronics in Agriculture**, v. 144, p. 269 – 283, 2018. ISSN 0168-1699. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0168169916312558>. Acesso em: 3 de dezembro de 2020. Citado na página 4.

7 KATH, J.; PEMBLETON, K. G. A soil temperature decision support tool for agronomic research and management under climate variability: Adapting to earlier and more variable planting conditions. **Computers**

and Electronics in Agriculture, v. 162, p. 783 – 792, 2019. ISSN 0168-1699. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0168169918317629>. Acesso em: 3 de dezembro de 2020. Citado na página 5.

8 TRILLES, S. et al. Development of an open sensorized platform in a smart agriculture context: A vineyard support system for monitoring mildew disease. **Sustainable Computing: Informatics and Systems**, 2019. ISSN 2210-5379. Disponível em: <http://www.sciencedirect.com/science/article/pii/S2210537918302270>. Acesso em: 3 de dezembro de 2020. Citado na página 5.

9 OLIVEIRA, B. **Dando uma breve análise no protocolo MQTT**. 2020. Disponível em: <https://medium.com/internet-das-coisas/iot-05-dando-uma-breve-an%C3%A1lise-no-protocolo-mqtt-e404e977fbb6>. Acesso em: 30 de março de 2022. Citado na página 6.

10 TEAM, T. H. **Introducing the MQTT Protocol - MQTT Essentials: Part 1**. Disponível em: <https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/>. Acesso em: 5 de abril de 2022. Citado nas páginas 6 e 7.

11 IBM Docs. 2022. Disponível em: <https://www.ibm.com/docs/en/ibm-mq/8.0?topic=overview-introduction-mq>. Acesso em: 6 de abril de 2022. Citado na página 6.

12 MILINKOVICH, M. **About the Eclipse Foundation | The Eclipse Foundation**. Disponível em: <https://www.eclipse.org/org/>. Acesso em: 6 de abril de 2022. Citado na página 6.

13 ABOUT Us. Disponível em: <https://www.oasis-open.org/org/>. Acesso em: 6 de abril de 2022. Citado na página 7.

14 MQTT. **Use Cases**. 2020. Disponível em: <https://mqtt.org/use-cases/>. Acesso em: 20 de abril de 2022. Citado na página 8.

15 HIVEMQ. **HiveMQ MQTT Broker**. 2020. Disponível em: <https://www.hivemq.com/hivemq/mqtt-broker/>. Acesso em: 20 de abril de 2022. Citado na página 8.

16 HIVEMQ. **Car-Sharing Application relies on HiveMQ for Reliable Connectivity**. 2020. Disponível em: <https://www.hivemq.com/case-studies/bmw-mobility-services/>. Acesso em: 20 de abril de 2022. Citado nas páginas 8 e 9.

17 HIVEMQ. **HiveMQ's Reliable IoT Communication Enables Real-time Monitoring of Matternet's Autonomous Drones**. 2020. Disponível em: <https://www.hivemq.com/case-studies/matternet/>. Acesso em: 20 de abril de 2022. Citado nas páginas 9 e 10.

18 DINCULEANă, D.; CHENG, X. Vulnerabilities and limitations of mqtt protocol used between iot devices. 2019. Citado nas páginas 17 e 18.

19 IBM. **Dealing with Distributed Denial of Service attacks**. 2019. Disponível em: <https://cloud.ibm.com/docs/cis?topic=cis-distributed-denial-of-service-ddos-attack-concepts>. Acesso em: 22 de abril de 2022. Citado na página 17.

20 REGAN, J. **O que é uma botnet e como você pode proteger seu computador?** 2018. Disponível em: <https://www.avg.com/pt/signal/what-is-botnet>. Acesso em: 22 de abril de 2022. Citado na página 17.

21 IBM. **Access control lists**. 2021. Disponível em: <https://www.ibm.com/docs/en/i/7.1?topic=security-access-control-lists>. Acesso em: 22 de abril de 2022. Citado na página 17.

22 IBM. **Conceitos de TLS (Transport Layer Security)**. 2021. Disponível em: <https://www.ibm.com/docs/pt-br/ibm-mq/9.0?topic=ssfksj-9-0-0-com-ibm-mq-sec-doc-q009920--htm>. Acesso em: 22 de abril de 2022. Citado na página 18.

23 IBM. **Protocolos de segurança criptográficos: TLS**. 2021. Disponível em: <https://www.ibm.com/docs/pt-br/ibm-mq/9.0?topic=ssfksj-9-0-0-com-ibm-mq-sec-doc-q009910--htm>. Acesso em: 22 de abril de 2022. Citado na página 18.

24 MICROSOFT. **Protocolo DTLS**. 2022. Disponível em: <https://www.ibm.com/docs/en/zos/2.1.0?topic=internets-tcp-udp-ip>. Acesso em: 22 de abril de 2022. Citado na página 18.

25 OASIS. **MQTT Version 5.0**. 2019. Disponível em: <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.pdf>. Acesso em: 22 de abril de 2022. Citado na página 18.

26 IANA. **The global coordination of the DNS Root, IP addressing, and other Internet protocol resources is performed as the Internet Assigned Numbers Authority (IANA) functions.** 2006. Disponível em: <https://www.iana.org/>. Acesso em: 22 de abril de 2022. Citado na página 18.