MQTT Protocol: Fundamentals, Tools and Future Directions

S. Quincozes, E. Tubino, and J. Kazienko

***Abstract*—Internet of Things (IoT) is a prominent paradigm applied to several areas ranging from medicine to industrial networks. It aims at connecting to the Internet of million of daily used objects. Important challenges in this area consist in to propose mechanisms and protocols that meet security, device interoperability, quality of service and energy-efficiency requirements. Particularly, the Message Queue Telemetry Transport (MQTT) protocol has been prospected in order to provide efficient communication at the application layer for the IoT. This survey aims to present the fundamentals, tools and future directions related to MQTT protocol and its variation tailored for sensor networks, called MQTT-SN. We discuss such protocols comparing to other current IoT application layer protocols, such as Constrained Application Protocol (CoAP). Additionally, we present tools so as to support practical experimentation and simulation. Particularly, we carry out pactical experiments to observe the communication delay between MQTT and CoAP. Finally, the open issues and challenges in this area are examined.**

***Index Terms*—Internet of Things, Application Layer, Survey, MQTT.**

I. Introdução

OS primeiros protótipos da tecnologia precursora da internet (*i.e.*, ARPANET) surgiram na década de 1960, com a finalidade de comunicação militar e acadêmica. Com o surgimento da Internet, em meados de 1980, o acesso a essa rede logo se popularizou mundialmente. Atualmente, o número de usuários e aplicações continua em constante crescimento. Desse modo, o futuro da Internet é motivo de discussão. Nesse contexto, novas tecnologias e abordagens vêm surgindo para atender as necessidades de comunicação de suas novas aplicações, tais como a comunicação entre máquinas, Internet orientada a conteúdo e a Internet das Coisas, do inglês, *Internet of Things* (IoT) [1] [2].

Particularmente, o paradigma IoT é promissor para a implantação de novas aplicações que conectam diversos dispositivos e objetos comuns do dia-a-dia à Internet, possibilitando a sua identificação e comunicação. Para tanto, diversas tecnologias de comunicação vêm sendo utilizadas, tais como, IEEE 802.15.4, IEEE 802.11 (WiFi), IEEE 802.15.1 (BLE), LoRa and SigFox. Todavia, devido à alta demanda por baixo consumo energético de muitos dos dispositivos da IoT, o escopo deste trabalho envolve os protocolos de comunicação aplicados à tecnologia IEEE 802.15.4. Nesse contexto, se fazem necessários mecanismos e protocolos adaptados para os requisitos impostos por tais dispositivos, principalmente em termos de economia de energia e processamento [3].

Atualmente, existe uma série de propostas de protocolos na literatura que visam permitir a comunicação de dados entre diferentes tipos de dispositivos, com distintas capacidades computacionais e disponibilidades energéticas. Para a camada de aplicação na IoT, em especial, alguns protocolos têm sido adotados, como o protocolo de aplicação para recursos limitados, do inglês, *Constrained Application Protocol* (CoAP) [4], protocolo de fila de mensagens para transporte de telemetria, do inglês, *Message Queuing Telemetry Transport* (MQTT) [5], e MQTT *Sensor Networks* (MQTT-SN) [6], adaptado para operar em redes de sensores. Embora não haja uma consolidação na escolha de um desses protocolos para uso geral, estudos indicam a proeminência do protocolo MQTT, especialmente em redes de sensores sem fio e na comunicação Veículo-a-Veículo (V2V) [7] [8] [9] [3].

Contudo, segundo Meena et. Al. [7], ambos os protocolos CoAP e MQTT apresentam limitações, especialmente, no que diz respeito a garantia da segurança da informação. Isso ocorre devido ao fato de que o uso de mecanismos convencionais, tais como TLS e SSL, não são apropriados para dispositivos com restrições de recursos energéticos e computacionais existentes na IoT. Ademais, as propostas alternativas, tais como, a autenticação baseada no uso de usuário e senha [5], não são eficientes por conta da falta de confidencialidade na comunicação. Desse modo, são necessários novos mecanismos a fim de prover segurança de maneira leve e eficiente [8].

O objetivo deste trabalho consiste em apresentar um estudo acerca dos protocolos MQTT e MQTT-SN a fim de identificar as vantagens e limitações de seu uso, principalmente, em relação aos protocolos CoAP e HTTP. Adicionalmente, são apresentadas ferramentas a fim de desenvolver experimentos práticos e simulações através do MQTT. Por fim, são discutidos os desafios de pesquisa e direções futuras da área.

O restante deste trabalho está organizado como segue. Na Seção II, o paradigma da IoT será brevemente introduzido. A Seção III apresenta as alternativas de protocolos para a camada de aplicação da IoT e suas características. A Seção IV descreve ferramentas que permitem a experimentação desses protocolos. A Seção V discute as vantagens e limitações dos protocolos apresentados anteriormente. Em seguida, os desafios de pesquisa e direções futuras são apresentados na Seção VI. Por fim, a Seção VII apresenta as conclusões.

II. Internet Das Coisas

A IoT é um paradigma aplicado a várias áreas, desde a medicina até as redes industriais que visiona a conexão

Silvio E. Quincozes is with the Computing Institute, Fluminense Federal University, Niterói, RJ, Brazil, e-mail: sequincozes@id.uff.br.

Emilio R. Tubino is with Computer Science Departament, Federal University of Pampa, Alegrete, RS, Brazil, e-mail: tubino@unipampa.edu.br.

Juliano F. Kazienko is with Industrial Technical College, Federal University of Santa Maria, Santa Maria, RS, Brazil, e-mail: kazienko@redes.ufsm.br.

de bilhões de objetos comuns do dia-a-dia à Internet, de modo a permitir a interação com os mesmos. Nesse contexto, os dispositivos envolvidos caracterizam-se, tipicamente, pelas restrições computacionais e energéticas. Desse modo, uma vez que a Internet convencional apresenta limitações em termos de número de dispositivos conectados e no suporte ao baixo poder energético dos dispositivos que fazem parte da IoT, novas abordagens e protocolos são necessários para que os mesmos sejam acessíveis pelas aplicações futuras por meio de uma conexão com a internet [3] [10].

A. Pilha de Protocolos da IoT

Existem muitas propostas de protocolos que podem constituir a arquitetura de redes da IoT. A Fig. 1 ilustra uma arquitetura de rede para IoT com sua respectiva pilha de protocolos, a qual remete aos protocolos que são comumente utilizados em aplicações que envolvem dispositivos limitados em termos de recursos computacionais como processamento, memória, banda e energia (bateria). Cabe ressaltar que existem outros modelos arquiteturais na literatura [3] [11].

A pilha de protocolos propostos para as aplicações do paradigma da IoT tem seu nível mais baixo padronizado pela especificação IEEE 802.15.4, que define as camadas física e de enlace. O padrão IEEE 802.15.4 também é a base para a especificação da tecnologia ZigBee, a qual é comumente utilizada para a comunicação em redes de sensores sem fio. Alternativamente, a camada física pode envolver outros protocolos tais como LTE-A, ECP-Global e Z-Wave [3].

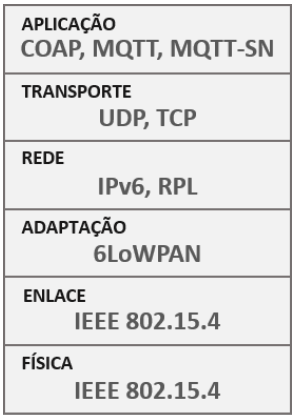

Fig. 1. Arquitetura de redes para a IoT, adaptado de [12].

Tradicionalmente, a Internet faz uso do Internet Protocol (IP), conhecido como protocolo IPv4, em sua camada de rede. Todavia, devido ao número de dispositivos previstos na IoT, que chegará na casa dos 40 bilhões em 2020 [7], sua camada de rede prevê o uso do protocolo IPv6, de modo a suportar a quantidade de endereços IPs requerida [3]. Porém, uma adaptação desse protocolo se faz necessária a fim de comunicar dados em baixa potência em redes de área pessoal sem fio, do inglês, *IPv6 over Low-Power Wireless Personal Area Networks* (6LoWPAN). Adicionalmente, a pilha de protocolos da IoT contempla um protocolo específico para o roteamento de mensagens em redes de baixa potência e com perdas. Esse protocolo é conhecido pela sigla provinda de sua nomenclatura em inglês: *Routing Protocol for Low Power and Lossy Networks* (RPL) [13].

Na camada de transporte, encontram-se os tradicionais protocolos User Datagram Protocol (UDP) e Transmission Control Protoco (TCP). Ambos os protocolos são utilizados, contudo, aplicações que optam pelo uso do UDP, devido a sua simplicidade, necessitam de tratativas adicionais para a garantia da confiabilidade na comunicação de mensagens.

Por fim, a camada de aplicação é composta por protocolos alternativos ao HTTP, tais como, CoAP, MQTT e MQTT-SN.

III. Protocolos de Camada de Aplicação

A camada de aplicação é responsável por prover serviços ao usuário final, portanto sua principal importância consiste na capacidade de fornecer tais serviços inteligentes com uma alta qualidade [3]. Assim, ao invés do uso do protocolo HTTP, adotado pela Internet tradicional, na IoT, a camada de aplicação demanda protocolos adequados à comunicação de dispositivos de baixo poder computacional [1] [8] [3].

A. Protocolo CoAP

O CoAP consiste em um protocolo de comunicação orientada a serviços, do inglês, *Service Oriented Architecture* (SOA), que se baseia nos princípios de transferência representacional de estado, do inglês, *Representational State Transfer* (REST) para a transferência de dados. Desse modo, serviços WEB podem oferecer o acesso, recuperação, modificação e exclusão de dados através de endereços representados por *Uniform Resource Identifiers* (URIs) [4].

O protocolo CoAP faz uso do UDP na camada de transporte, permitindo que dispositivos de baixo poder computacional possam implementar e usar o CoAP [14]. Contudo, embora haja redução de complexidade, o uso de UDP exige a implementação de medidas adicionais para a detecção de mensagens duplicadas e garantia de confiabilidade [1] [15]. Ademais, são trocados sete tipos de mensagens no CoAP. Os quatro principais tipos são apresentados a seguir:

1) Confirmable: utilizado para a confirmação de recebimento de mensagens sem a ocorrência de perda de pacotes;

2) Non-Confirmable: utilizado para a transmissão de mensagens que não requerem nenhuma confirmação de entrega (e.g. leituras consecutivas de sensores);

3) Acknowledgement Message: são mensagens que indicam a confirmação do recebimento de mensagens *Confirmable*;

4) Reset: indica que alguma mensagem, seja *Confirmable* ou *Non-Confirmable*, foi recebida, mas não foi possível sua total interpretação (*e.g.*, após reinicialização do nó).

Embora o CoAP tenha sido projetado em torno de arquiteturas bem estabelecidas como o HTTP, algumas funcionalidades não são contempladas em sua especificação original. Portanto, muitas vezes, se faz necessário o uso de extensões para complementar a sua implementação incluindo funcionalidades, tais como, descoberta de serviços e economia de energia [10].

No entanto, existem extensões que podem tornar o uso do CoAP adequado para aplicações diferentes daquelas suportadas originalmente. O *Internet-Draft* [16] estende o

CoAP, habilitando a comunicação através do paradigma publicação/inscrição, tal como é feito no MQTT. Desse modo, o CoAP é adaptado para uso em dispositivos que ficam desconectados por longos períodos. Contudo, as especificações existentes ainda são preliminares e requerem melhorias.

B. Protocolo MQTT

Diferentemente da abordagem de comunicação adotada por grande parte dos protocolos da Internet, baseada no paradigma Cliente-Servidor, o protocolo MQTT é baseado em um paradigma que permite a transmissão de mensagens para agrupamentos específicos de clientes de forma intermitente. Esse paradigma, conhecido como *Publish-Subscriber*, ou Publicador-Assinante, permite que clientes interessados passem a assinar tópicos de seu interesse em um servidor centralizado chamado *Broker* MQTT. Um *Broker* pode conter múltiplos tópicos, onde cada um deles permite a recepção de mensagens de diferentes dispositivos publicadores e as entrega aos múltiplos dispositivos assinantes daquele tópico.

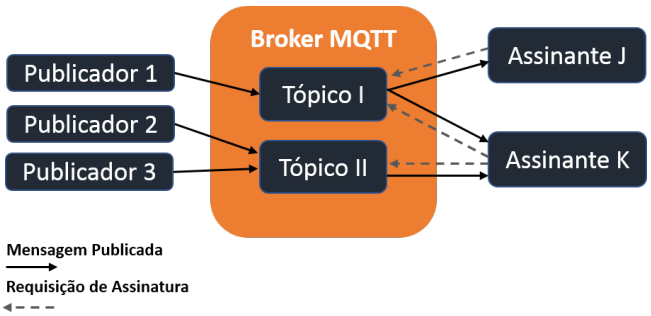

Fig. 2. Comunicação por meio do paradigma Publicador-Assinante adotado pelo protocolo MQTT, adaptado de [5].

A Fig. 2 ilustra um cenário hipotético onde três dispositivos publicadores alimentam um *broker* que possui dois tópicos e dois assinantes. Neste cenário, o dispositivo Assinante J manifesta interesse nas mensagens provindas do Tópico A, o qual é responsável por propagar as mensagens provindas do dispositivo Publicador 1. O Assinante K, por sua vez, além de receber os dados publicados pelo dispositivo Publicador 1, por meio da assinatura ao Tópico A, recebe os dados publicados pelo Publicadores 2 e 3, por meio da assinatura ao tópico B. Desse modo, sempre que um dispositivo publicador envia uma mensagem para o *Broker*, tal mensagem é propagada para todos os dispositivos assinantes. Portanto, é possível a comunicação de forma assíncrona, não necessitando que publicadores e assinantes estejam ativos em simultâneo para que ocorra a comunicação entre os mesmos.

Portanto, tal como o CoAP, o MQTT é mais leve que o protocolo HTTP. Desse modo, esse é um protocolo apropriado para aplicações IoT que possuem restrições no consumo de recursos computacionais e de rede [8] [15]. Devido ao seu paradigma de comunicação ser baseado em Publicação-Assinatura, o MQTT é mais adequado para aplicações que requerem maior escalabilidade e economia de energia por parte de dispositivos. Ademais, diferentemente do CoAP, as especificações do protocolo MQTT não definem um protocolo da camada subjacente específico, mas especifica os requisitos que esse protocolo deve cumprir (*e.g.*, clientes e servidores devem suportar o uso de um ou mais protocolos de transporte que forneçam um fluxo de bytes ordenado e sem perdas). Assim, o protocolo TCP é comumente adotado, uma vez que o mesmo atende tais exigências [10] (em contraste com o MQTT-SN, cuja a especificação é adaptada para dispositivos simples, e permite a adoção do protocolo UDP [7]).

O protocolo envolve o uso de 14 tipos de mensagens [5]. Dentre essas mensagens, as mais relevantes são aquelas envolvidas nos processos de publicação de mensagens, inscrição nos tópicos de interesse e cancelamento de assinaturas aos tópicos indesejados, conforme listado a seguir:

1) Publish: consistem em mensagens enviadas por um cliente publicador para um servidor, chamado *Broker*, o qual tem a finalidade de efetuar sua propagação aos demais clientes assinantes do respectivo tópico;

2) Subscribe: esse tipo de mensagens permite que um cliente registre interesse em um ou mais tópicos no servidor. Assim, as mensagens publicadas em tópicos assinados serão recebidas do servidor;

3) Unsubscribe: mensagens desse tipo permitem o cancelamento da assinatura aos tópicos indesejados.

Em geral, as mensagens MQTT contém cabeçalhos de tamanho fixo de oito bits, quais sejam: tipo da mensagem, indicador de duplicidade, nível de QoS e indicador de retenção. Esse último bit tem a finalidade de manter a mensagem no *broker* após a entrega — exclusivamente em mensagens *publish*. Além disso, algumas mensagens requerem um componente adicional chamado cabeçalho variável e uma carga útil. Essas informações ocupam o espaço de pelo menos um byte. Nas mensagens *Connect*, por exemplo, o cabeçalho variável contém dados como o nome e versão do protocolo, enquanto que na carga útil estão presentes informações pertinentes ao método de autenticação, incluindo o identificador único do cliente, nome de usuário e senha. É importante observar que se o uso de TLS não for adotado, essas informações são transmitidas em texto plano [5].

C. Protocolo MQTT-SN

O protocolo MQTT-SN consiste em uma implementação do MQTT adaptada para operar sob as peculiaridades das redes de sensores sem fio, a qual prevê comunicação entre dispositivos de baixo poder computacional e com a possibilidade de permanecer inativos por determinados períodos de modo a economizar energia.

Assim como o MQTT tradicional, o MQTT-SN se baseia no paradigma de publicação e assinatura. Para tanto, os clientes MQTT-SN que desejam fazer o envio de mensagens ou inscrição a tópicos estabelecem comunicação com gateways capazes de efetuar a conversão de mensagens a partir de múltiplas interfaces de rede pelo protocolo MQTT-SN para o *broker* MQTT tradicional, permitindo assim a interoperabilidade entre diferentes tipos de dispositivos [7].

Ademais, o protocolo MQTT-SN provê a capacidade de gerenciamento de tempo de atividade do rádio de dispositivos de uma rede por meio da inclusão de uma duração

nas mensagens *DISCONNECT*, possibilitando que o *gateway* armazene temporariamente as mensagens durante esse período. Ao contrário do MQTT tradicional, onde os dispositivos assinantes gerenciam suas próprias conexões com o *broker*, no MQTT-SN a conexão com o *broker* é gerenciada pelo *gateway*. Desse modo, o consumo de energia é concentrado nesse dispositivo que tipicamente possui menores restrições energéticas que os demais clientes [7]. Entretanto, a comunicação entre o *gateway* e os clientes pode ser insegura, se não tratada pela aplicação. A Fig. 3 ilustra o funcionamento do protocolo MQTT-SN.

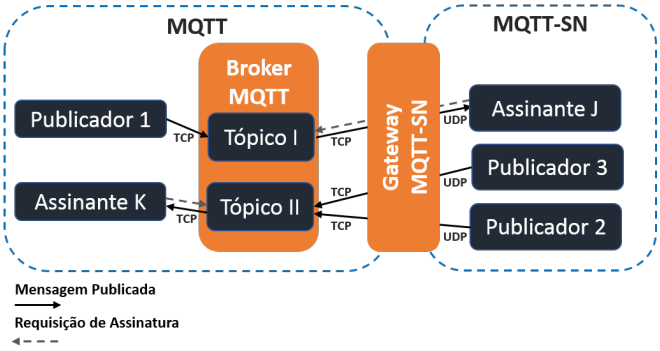

Fig. 3. Paradigma publicador-assinante adotado pelo protocolo MQTT-SN, adaptado de [17].

Suponha um cenário industrial onde o sistema de refrigeração é capaz de se ajustar de acordo com a temperatura das máquinas em funcionamento. Esse cenário pode ser implementado com o uso de um conjunto de sensores e atuadores, conforme ilustrado na Fig. 4. Os sensores, implantados nas máquinas, são responsáveis pela leitura e publicação das temperaturas (passo 1.1). Tais sensores podem se comunicar através de uma tecnologia sem fio de múltiplos saltos, onde a mensagem é encaminhada até chegar no gateway MQTT-SN (passo 1.2). Em seguida, o gateway converte a mensagem para o formato MQTT e a encaminha para o *Broker* MQTT (passo 2), que a transmite aos assinantes do tópico (passo 3).

Ao receber e processar as informações de temperatura das máquinas, o dispositivo de controle verifica que a temperatura está alta e, portanto, publica uma mensagem para um conjunto de atuadores, que estão acoplados nos aparelhos de refrigeração. Essa mensagem é encaminhada ao *broker* MQTT (passo 4) e então encaminhada para os dispositivos assinantes daquele tópico por meio do gateway MQTT-SN (passo 5). O gateway MQTT-SN entrega a mensagem, diretamente ou através de nós intermediários (passos 6.1a e 6.1b) para todo o conjunto de dispositivos assinantes (passos 6.2a e 6.2b) que devem reduzir a temperatura do aparelho refrigerador, conforme os parâmetros recebidos.

O emprego do gateway MQTT-SN possibilita a interoperabilidade de dispositivos internos e externos à rede de sensores. Portanto, diferentemente do CoAP, essa versão do protocolo MQTT estabelece a comunicação não só entre diferentes dispositivos de um mesmo tipo de rede, mas também promovem a comunicação transparente entre dispositivos que utilizam diferentes tecnologias de comunicação [17].

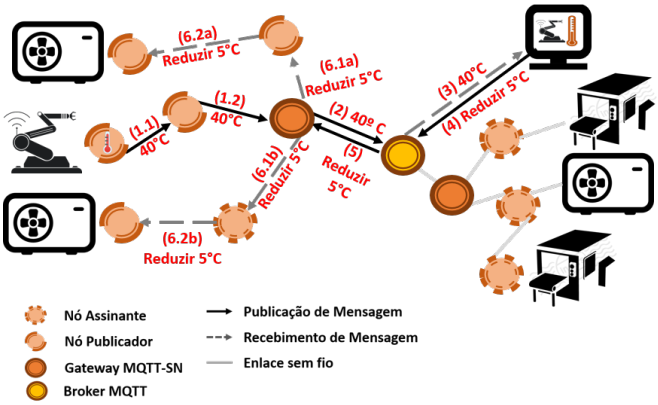

Fig. 4. Uso do MQTT-SN em cenário industrial.

D. Outros Protocolos

Embora os protocolos discutidos anteriormente sejam mais citados na literatura, existem alternativas que podem ser adotadas para o uso na camada de aplicação da IoT.

O protocolo de Serviço de Distribuição de dados, do inglês, *Data Distribution Service* (DDS) adota paradigma de comunicação Publicador/Assinante (P/A), como no MQTT. Porém, ao invés de uma arquitetura centralizada, no DDS a arquitetura é descentralizada. Na camada de transporte são suportados ambos os protocolos UDP e TCP. Estudos demonstram que o uso de CPU tem comportamento linear no MQTT enquanto cresce exponencialmente no DDS [2]. Isso ocorre devido a abordagem descentralizada, onde os publicadores necessitam enviar suas mensagens diretamente para cada um dos assinantes. Esse cenário é inviável para dispositivos com restrições energéticas, contudo mitiga o problema do ponto único de falhas. Já na abordagem centralizada, adotada pelo MQTT, o publicador precisa transmitir somente uma mensagem para o *Broker*, que usualmente possui recursos como alto poder computacional e uma fonte de energia [2]. Arquiteturas centralizadas, no entanto, podem estar susceptíveis a um ponto único de falhas.

Outro protocolo alternativo para uso na camada de aplicação da IoT é o protocolo avançado de enfileiramento de mensagens, do inglês, *Advanced Message Queuing Protocol* (AMQP). O protocolo AMQP possui três níveis de QoS, tal como o MQTT. Na camada de transporte, é usado o protocolo UDP, assim como o CoAP. Todavia, esse protocolo não garante interoperabilidade e não provê bibliotecas de código aberto para dispositivos com restrições computacionais [2].

O protocolo de mensagens e presença extensível, do inglês, *Extensible Messaging and Presence Protocol* (XMPP) possui uma autenticação baseada em SASL, codificação UTF-8 e suporta criptografia SSL/TLS. Ainda, esse protocolo apresenta boa interoperabilidade devido ao uso do formato *eXtensible Markup Language* (XML). Contudo, algumas limitações tornam seu uso inviável em dispositivos da IoT, tais como, o alto consumo de CPU e largura de banda. Além disso, não é oferecida nenhum suporte a QoS [2].

IV. Ferramentas de Experimentação

Uma das principais direções para a avaliação de protocolos para a IoT consiste no uso de ferramentas destinadas a implementações práticas ou simulações desses protocolos.

A. Ferramentas para Experimentos Práticos

A maneira mais realística de avaliar os diferentes protocolos é a experimentação prática. Para tanto, existem algumas opções de implementações disponíveis, em diferentes linguagens e com diferentes variações.

A ferramenta Mosquitto consiste em um pacote disponível para o Ambiente de Desenvolvimento Integrado, do inglês, Integrated Development Environment (IDE) Eclipse que implementa o protocolo MQTT de maneira prática. Esse pacote possibilita a hospedagem de um *broker* MQTT em um dispositivo físico onde os clientes são capazes de realizar as operações de leitura e escrita. Portanto, o Mosquitto pode ser utilizado tanto para experimentos práticos da comunicação MQTT em dispositivos e ambientes virtuais, quanto no desenvolvimento de aplicações com dispositivos e meios de comunicações reais. Essa ferramenta é disponibilizada em [18]. Junto a ela, a Interface de Programação de Aplicação, do inglês, Application Programing Interface (API) chamada Paho [19] poder ser utilizada para a implementação de clientes MQTT ou MQTT-SN. Adicionalmente, existe um servidor publicamente acessível para os projetos do Eclipse IoT disponível em *iot.eclipse.org*, porta 1883 [19]. Similarmente, a biblioteca *HiveMQ* [20] permite a implementação de um cliente em Java e disponibiliza um servidor público em *broker.hivemq.com:1883*.

Outra ferramenta é a Moquette [21], que assim como a Mosquito, é um pacote disponível para o Eclipse, que permite configurar a comunicação MQTT ao utilizá-la. Essa ferramenta é leve e de fácil entendimento o que proporciona uma maior facilidade em sua incorporação em um projeto.

Outro protocolo da camada de aplicação, o CoAP possui uma implementação na linguagem Java disponível chamada Californium (Cf) CoAP framework. O servidor do CoAP pode ser implementado por meio da ferramenta Cf core [22]. Ainda, a biblioteca de código aberto *CoapBlaser* [23] disponibiliza tanto a implementação de cliente quanto a de servidor CoAP. Por outro lado, o protocolo DDS pode ter seus nós implementados por meio da ferramenta RTI Connext Java API. Já os protocolos XMPP e AMQP podem ter seus dispositivos clientes usando as APIs Smack e Java AMQP, respectivamente, enquanto os servidores podem ser implementados por meio do OpenFire e RabbitMQ.

B. Ferramentas para Simulações

De forma geral, os simuladores imitam uma situação real ou hipotética em um computador. Um dos simuladores mais populares voltado para a IoT em geral é *Cooja* [24], o qual se baseia no sistema operacional *Contiki OS*, para simulação de redes de sensores e objetos inteligentes por meio da emulação de sensores reais. Por meio dessa ferramenta é possível o desenvolvimento e execução de aplicações que suportam o protocolo MQTT nos sensores que são emulados. Sua principal utilização se dá na execução de testes em grandes escalas, onde o número de sensores em uma rede é relativamente alto. A emulação de sensores suporta dispositivos heterogêneos.

Portanto, uma das principais vantagens de ambientes de simulação como esse se dão pela capacidade de tornar o processo de avaliação experimental mais barato, dispensando a necessidade da aquisição de dispositivos reais. Além disso, possibilita testes em maior escala. Outro aspecto importante é que o protocolo MQTT pode ser facilmente desenvolvido e aplicado em simulações por conta do código-fonte dos ambientes e dos sensores emulados serem abertos.

Vale ressaltar que já é possível encontrar algumas implementações do MQTT e MQTT-SN para o Contiki feitas pela comunidade e que estão disponíveis em projetos abertos no GitHub. Algumas dessas implementações possibilitam o uso da ferramenta Mosquitto, no papel de borker, em conjunto com o Cooja que simula os clientes: um exemplo é o projeto *homestark_mqtt_6lowpan_port* [25]. No GitHub é possível também encontrar exemplos e tutoriais de como rodar um servidor CoAP na ferramenta Contiki [19].

O principal propósito do Cooja consiste em possibilitar a simulação de redes IoT, permitindo a captura de uma série de dados resultantes da simulação, tais como o consumo de energia de cada um dos sensores para a execução de determinadas tarefas. Assim, tais simulações permitem a comparação de diferentes variações de implementação (e.g. protocolo MQTT e MQTT-SN) ou uso de diferentes protocolos da camada subjacente (e.g. UDP e TCP). Por fim, os códigos gerados e simulados no Contiki OS podem ser exportados e testados em ambientes com dispositivos reais.

C. Resultados Comparativos

Embora o escopo deste trabalho se concentre na revisão comparativa das propriedades dos protocolos aplicáveis à camada de aplicação da IoT, a seguir é apresentada uma experimentação prática a fim de mensurar os tempos médios de respostas para publicações no protocolo MQTT e o envio de requisições no protocolo CoAP.

Para a avaliação do protocolo MQTT, foram consideradas a publicação de 500 mensagens a partir de um dispositivo publicador e um *broker* publicamente disponível [20] . O dispositivo publicador foi implementado através da biblioteca HiveMQ em smparthone Android Huawei Honor 8x, com 4GB de RAM e processador Octa Core (4 de 2.2GHz + 4 de 1.7GHz). A publicação de mensagens é direcionado ao *broker* disponibilizado publicamente no endereço *broker.hivemq.com*, na porta 1883.

O protocolo CoAP foi avaliado com base na implementação do cliente por meio da biblioteca *CoapBlaster* [23] no mesmo dispositivo usado como publicador no experimento anterior. Neste cenário, tal dispositivo envia 500 requisições a um servidor CoAP, o qual encontra-se publicamente disponível no endereço *coap://coap.me/test*.

Para ambos os experimentos, verificou-se os hospedeiros que executam o servidor CoAP e o *Broker* MQTT são providos pelo serviço *Amazon AWS* e estão localizados fisicamente na Alemanha. O dispositivo *smartphone* usado como cliente

CoAP e publicador MQTT localiza-se no Brasil. O número preciso de saltos entre as partes não foi retornado pela ferramenta *traceroute* devido ao bloqueio dessa informação por parte dos hospedeiros. Similarmente, não foi possível verificar a latência com a ferramenta *ping*. Contudo, com base na localização, descrita acima, e nos serviços de infraestrutura utilizados, acredita-se que não são introduzidos atrasos significativos nos experimentos realizados. Ademais, ambas as coletas baseiam-se na média a partir de 500 amostras.

Os resultados comparativos são exibidos nos gráficos apresentados nas Figs. 5 e 6, onde são apresentados os tempos de cada amostra e o tempo médio de todas as amostras, respectivamente. Na Fig. 6, foi considerado um intervalo de confiança de 95%. Com base nos dados apresentados nessa figura, é possível notar que o MQTT no nível 2 de QoS apresenta uma sobrecarga significativa no cenário experimentado — onde o *broker* encontra-se fisicamente distante do publicador.

Em relação ao MQTT - QoS 1, observa-se na Fig. 6 que os tempos de resposta são estatisticamente equivalentes ao do protocolo CoAP. Ou seja, tais protocolos são similares em termos de atrasos. Cabe ressaltar que, para o MQTT, o tempo contabilizado consiste no intervalo entre a mensagem PUBLISH e sua confirmação, PUBACK. Desse modo, o tempo de estabelececimento de conexões não é contabilizado. Essa metodologia visa uma medição justa, dado que o CoAP não é orientado a conexão devido ao uso do protocolo UDP. Além disso, em um cenário real, múltiplas publicações podem ocorrer a partir de uma mesma conexão.

Por fim, o atraso médio para o QoS nível 0 do MQTT é estimado como sendo metade do valor do QoS nível 1. Essa estimativa leva em conta que (i) não é possível medir o valor exato devido a ausência de mensagem de confirmação (*i.e.,* o nível de QoS 0 não possui a mensagem PUBACK) e (ii) supõe-se que o tempo para envio da mensagem de *PUBLISH* seja aproximado do tempo de recebimento da mensagem *PUBACK*.

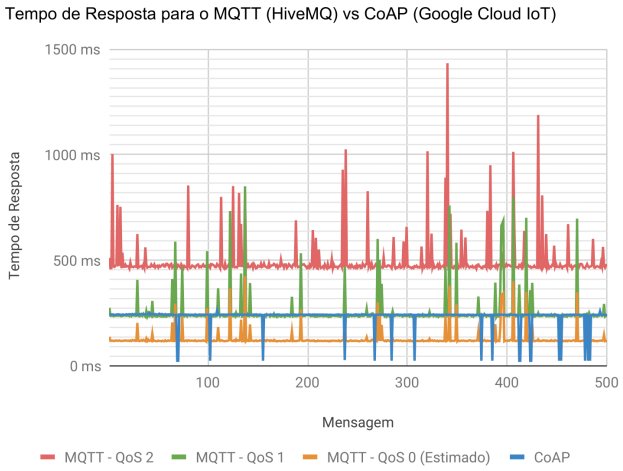

Fig. 5. Tempo de resposta para cada mensagem transmitida, excluso tempo de conexão no MQTT.

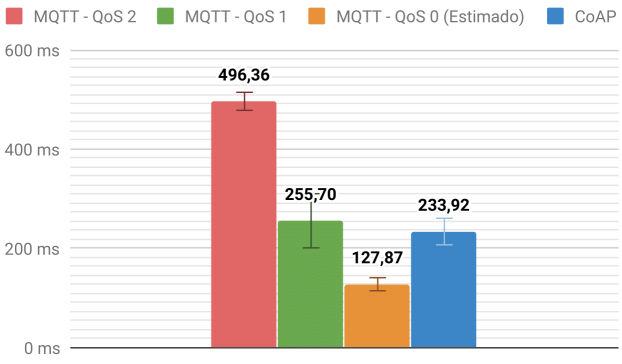

Fig. 6. Tempo de resposta médio para o envio de 500 mensagens, excluso tempo de conexão no MQTT.

V. Discussão

Diversos protocolos foram propostos para a camada de aplicação da IoT. Assim, se faz necessária a compreensão das particularidades de cada um desses protocolos e sua conveniência para cada aplicação [10]. Esta Seção discute tais questões, apresentando as vantagens e limitações no uso dos protocolos MQTT, MQTT-SN e outras alternativas existentes.

A. Segurança

Um dos fatores determinantes para a escolha do protocolo de aplicação consiste nas propriedades de segurança que o mesmo provê. A seguir, tais propriedades serão discutidas.

Confidencialidade: a alta restrição de capacidade computacional de dispositivos miniaturizados, especialmente aqueles de classe zero – onde as memórias RAM e flash disponíveis são inferiores a, respectivamente, 10 e 100KB – torna impraticável a maioria das medidas de segurança aplicáveis na computação tradicional. As especificações dos protocolos MQTT, MQTT-SN e CoAP assumem o uso de mecanismos tradicionais tais como SSL/TLS e DTLS, com gerência de certificados e chaves dentre os dispositivos. Contudo, tais mecanismos requerem um alto custo computacional, muitas vezes não sendo suportado por dispositivos da IoT com recursos restritos. Adicionalmente, o uso de certificados e o gerenciamento de chaves de sessão para todos os dispositivos pode se tornar inviável devido a escala e heterogeneidade de dispositivos [7].

Disponibilidade: uma das desvantagens eminentes em qualquer arquitetura que contém uma entidade centralizadora consiste na existência de um ponto único de falhas. Segundo a especificação do protocolo MQTT, o *broker* de uma aplicação deve ser responsável por intermediar a comunicação entre os clientes que assinam ou publicam em seus tópicos. Assim, apesar de sua capacidade de processamento ser superior, os *brokers* ainda representam pontos únicos de falhas que podem comprometer o sistema em caso de invasões ou falhas.

Autenticidade: existem três formas de autenticação de clientes no protocolo MQTT: uso de nome de usuário e senha, uso de um número pessoal de identificação, do inglês, Personal Identification Number (PIN) e autenticação por meio do mecanismo SSL [5]. No entanto, a autenticação por meio de técnicas que envolvem credenciais (usuário e senha ou PIN) é

vulnerável à ataques baseados na captura dessas informações, pois não contempla confidencialidade [8]. A autenticação baseada em SSL, apesar de contemplar confidencialidade, aparentemente não é adequada para o uso em dispositivos de poder computacional restrito.

Ademais, existem diversos *brokers* que não efetuam nenhuma autenticação de seus clientes, o que não é obrigatório segundo a especificação. A ferramenta de análise de segurança *Shodan* [26], permite a visualização desses dispositivos vulneráveis. O filtro "*port:1883 MQTT Connection Code: 0*", por exemplo, é capaz de revelar informações tais como o endereço IP e nomes de tópicos de *brokers* configurados para operar na porta padrão, sem o uso do mecanismo TLS, que possuem código de conexão com valor zero (não autenticam seus clientes) [8].

Segundo o relatório extraído da ferramenta Shodan [26], em julho de 2018, foram mapeados um total de 35.100 dispositivos como resultado para o filtro aplicado. Esse número representa 67,28% do total de dispositivos mapeados com o filtro "port:1883 MQTT". A maior ocorrência de dispositivos com essa vulnerabilidade se dá principalmente nos países China (8.718 dispositivos) e Estados Unidos (4.935), onde concentram-se 40,26% do total de dispositivos mapeados. Além disso, nos últimos 14 meses houve um incremento de 108,68% no número de dispositivos configurados na porta padrão do MQTT para comunicação sem TLS (1883) presentes nos resultados dos relatórios apresentados pela ferramenta Shodan, passando de 24.998 (27 de abril de 2017) para 52.168 (21 de julho de 2018).

Integridade: conforme reportado em [8], o protocolo MQTT é vulnerável a ataques capazes de executar a manipulação de dados de modo a comprometer a sua integridade. Para a prática desse tipo de ataque, é necessário o "envenenamento ARP", técnica na qual o atacante manipula a conexão de rede para que o tráfego legítimo passe por uma máquina maliciosa, onde as mensagens podem ser manipuladas a partir da sua identificação com base na aplicação de filtros e, posteriormente, adulteração dessas [8].

B. Eficiência Energética

O MQTT é apropriado para dispositivos de conexão intermitente, como é o caso de muitos dispositivos da IoT que desligam suas interfaces de rede a fim de economizar energia. Um dos fatores que reduz o consumo é o uso do paradigma de comunicação P/A, que implica na troca de um número reduzido de mensagens.

Por outro lado, o uso do protocolo TCP na camada de transporte de maneira subjacente ao MQTT acarreta em um número superior de mensagens do que implementações baseadas em UDP. Uma vez que o protocolo MQTT não conta com especificações diretas que exijam sua implementação sobre o protocolo de transporte TCP, essa não é uma desvantagem intrínseca do MQTT. Assim, a eficiência energética depende de uma série de fatores, tais como o tamanho e quantidade de mensagens que são transmitidas por cada cenário de aplicação.

Estudos de caso explorados através de experimentos realizados pelos autores do trabalho [9] demonstram que a transmissão de mensagens relativamente grandes (pacotes com tamanho superior a 1024 bytes) apresenta um consumo energético menor no uso do protocolo MQTT em relação ao CoAP. No entanto, é importante observar mensagens desse tamanho são atípicas em redes de sensores com recursos limitados. A versão do MQTT adaptada para redes de sensores (MQTT-SN), por sua vez, apresenta consumo energético inferior ao MQTT tradicional para a transmissão de mensagens de menor tamanho (até cerca de 512 bytes).

Outro aspecto favorável do uso do MQTT e MQTT-SN consiste no paradigma adotado (*i.e.*, P/A), o qual permite que publicadores e assinantes possam desligar suas interfaces de rádio sem sofrer perda de mensagens. Isso é alcançado devido a presença do dispositivo *Broker*, que permite o desacoplamento temporal entre as partes. Portanto, quando o assinante restabelecer sua interface de rádio, pode receber as mensagens transmitidas anteriormente pelo publicador. Em particular, o protocolo MQTT-SN facilita esse processo por meio de um procedimento chamado *keep-alive*. Tal funcionalidade permite que dispositivos entrem em um estado chamado modo *sleeping* por um determinado intervalo de tempo, onde o *broker* passa a armazenar todas as mensagens destinadas ao mesmo. Concretamente, para entrar nesse modo, o dispositivo deve enviar uma mensagem *DISCONNECT* para o *gateway* informando um valor no campo *DURATION*. Esse campo é opcional e quando não preenchido implica na finalização imediata da conexão.

C. Qualidade de Serviço

No protocolo MQTT são estabelecidos três diferentes níveis de qualidade de serviço, do inglês, Quality-of-Service (QoS). O nível 0 de QoS não considera a execução de retransmissões de mensagens, implicando assim na não garantia da entrega das mesmas. Portanto, nesse nível a entrega pode ocorrer uma única vez ou ainda não ocorrer [5].

Já para as mensagens com QoS nível 1, existe a garantia de que a entrega acontecerá. Contudo, não há um controle de entregas duplicadas. Portanto, as mensagens com esse nível de QoS podem ser entregues no mínimo uma e no máximo *N* vezes, onde *N* é o número de retransmissões efetuadas [5].

Por fim, o maior nível de qualidade de serviço provido pelo MQTT consiste no QoS de nível 2. Assim como no nível 1, as mensagens de QoS nível 2 também possuem garantia de entrega. Porém, nesse nível a retransmissão em duplicidade é eliminada. Assim, no QoS nível 2, a mensagem sempre é entregue exatamente uma vez única vez [5].

No protocolo MQTT-SN, quando utilizado o protocolo UDP na camada de transporte, a falta de confiabilidade nessa camada deve ser compensada na camada de aplicação. Para tanto, de modo similar ao MQTT, são estabelecidas classes predefinidas de qualidade de serviço. As mensagens com QoS nível 0 são enviadas de forma não confiável. O nível 1 de QoS se baseia em uma abordagem de parada e espera. Para tanto, o método de pedido de repetição automática, do inglês, *Automatic Repeat Request* (ARQ) é utilizado. Caso essa mensagem de retorno não seja recebida depois de um tempo predefinido, a mensagem deve ser retransmitida. Esse passo ocorre até que

o recebimento da mensagem seja confirmado pelo receptor ou que um número limite de tentativas seja excedido. O nível 2 de QoS garante a entrega de mensagens exatamente uma vez, onde o receptor deve informar o transmissor acerca do recebimento de mensagens íntegras por meio de uma mensagem de reconhecimento adicional. Entretanto, devido à sobrecarga ocasionada por esse nível de QoS, o mesmo não é adequado do ponto de vista de consumo energético e, portanto, não é recomendado no MQTT-SN [15]. Ademais, o MQTT-SN introduz um novo nível de QoS chamado *QoS -1*. Nesse nível, não há conexão, registro ou configuração de assinatura. O cliente só envia suas mensagens *PUBLISH* para um gateway com endereço conhecido *a priori* e, em seguida, efetua o descarte das mesmas — o descarte ocorre independentemente do sucesso na entrega da mensagem ao *gateway*.

D. Aplicações

A escolha do protocolo ideal deve considerar as diferentes características e restrições impostas pelos cenários nos quais as aplicações são empregadas. O protocolo MQTT pode ser considerado como uma das melhores opções quando consideradas suas capacidades em termos de uso de recursos, desempenho e confiabilidade. Conforme apresentado anteriormente, na Fig. 5, o MQTT - Qos 0 apresenta um tempo inferior ao CoAP em aplicações onde não há necessidade de confiabilidade na entrega das mensagens. Ademais, devido ao paradigma de comunicação no qual se baseia o MQTT, com o uso desse protocolo os dispositivos podem se comunicar de maneira assíncrona. Com isso, não é necessário que os dispositivos permaneçam ativos a todo momento ou em simultâneo. Essa funcionalidade é melhor explorada no MQTT-SN, pois o tempo de restabelecimento de conexão é reduzido, já que o *gateway* MQTT-SN gerencia as conexões dos dispositivos.

Portanto, embora que os resultados obtidos pelas experimentações de Mun et al. [9] tenham revelado um consumo energético mais eficiente no CoAP para mensagens de pequeno tamanho, acredita-se que a exploração da funcionalidade *sleeping* seja promissora para a economia de energia, mesmo para mensagens com níveis de QoS superiores — o que pode ser crucial para dispositivos com fonte limitada de energia.

Com isso, os protocolo MQTT e MQTT-SN têm seu uso bastante apropriado em áreas como aplicações industriais e redes de sensores, que exigem características como eficiência energética, confiabilidade e escalabilidade [7] [8] [9]. Em aplicações industriais, especialmente, a comunicação se torna custosa quando calcada no paradigma de comunicação Cliente-Servidor. Portanto, a arquitetura baseada em publicação e assinatura do MQTT pode reduzir o uso de recursos, em relação ao CoAP [10]. Segundo Muneer et. Al. [14], em aplicações de alto volume tráfego, o protocolo MQTT supera o protocolo CoAP, fornecendo maior vazão e menor latência. Essa afirmação é confirmada nos resultados dos experimentos de Collina et al. [27], que concluem que o MQTT oferece maior vazão e menor latência do que o CoAP em cenários de alta quantidade de tráfego, com alta porcentagem de perda e atraso de pacotes. Todavia, segundo Barroso et al. [2], o MQTT não é apropriado para aplicações que possuem altas taxas de amostragem. Ainda, resultados experimentais de Thangavel et al. [28] revelam que o MQTT apresenta um atraso menor do que o CoAP em cenários com baixa taxa de perda de pacotes.

O protocolo CoAP, por sua vez, é proeminente em aplicações como a Web das Coisas, do inglês, *Web-of-Things* (WoT), pois representa uma alternativa para o acesso a serviços web em dispositivos de baixo poder computacional com menor sobrecarga, comparado ao protocolo HTTP [29].

Segundo De Caro et al. [29], nos cenários com alta taxa de perda de pacotes, o CoAP apresentou menor atraso. Além disso, em cenários onde é necessária a confiabilidade na transmissão de mensagens, o envio de mensagens *CONFIRMABLE*, no CoAP, demonstra um atraso menor comparado ao MQTT com QoS 1 e 2. Entretanto, um ponto negativo do CoAP se dá pela necessidade do envio de mensagens *CON*, periodicamente, para garantir que o dispositivo cliente ainda está ativo. Essa é uma característica herdada do protocolo de camada de transporte adotado, que não mantém conexão (*i.e.*, UDP) [29].

É importante observar que, segundo os experimentos de De Caro et al. [29], mesmo com um protocolo de transporte mais simples, o CoAP demonstrou atrasos maiores para a entrega de mensagens *NON-CONFIRMABLE*, comparadas às mensagens de QoS nível 0 do MQTT. Então, segundo os autores, deve-se escolher o CoAP em cenários com alta taxa de perdas e o MQTT em cenários com baixa taxa de perdas.

Embora o CoAP não seja originalmente projetado para operar na comunicação de dispositivos em larga escala com a entrega de mensagens para grupos específicos de dispositivos, as extensões que vêm sendo desenvolvidas, tal como apresentado no internet draft [16], podem tornar seu uso adequado para novos cenários, incluindo aqueles que atualmente o MQTT e MQTT-SN tem maior proeminência.

A Tabela 1 apresenta a comparação entre os protocolos CoAP, MQTT, MQTT-SN e o tradicional HTTP.

TABELA I
COMPARAÇÃO ENTRE PROTOCOLOS DE APLICAÇÃO PARA IOT

	CoAP	**MQTT**	**MQTT-SN**	**HTTP**
Transporte	UDP	TCP	TCP/UDP	TCP
Comunicação	C/S	P/A	P/A	C/S
Segurança	DELS	SSL	SSL/TLS	SSL
Fragmentação	Custosa	Leve	Leve	-
QoS	2 níveis	3 níveis	4 níveis	-
Eficiência Energética	Média	Média	Alta	Baixa

Os pontos comparados consistem no protocolo utilizado na camada de transporte, paradigma de comunicação, classificado como Publicador/Assinante (P/A) ou Cliente/Servidor (C/S), mecanismo de segurança, robustez à fragmentação de pacotes, qualidade de serviço (QoS) e eficiência energética. Os protocolos listados, usualmente, envolvem o uso do protocolo TCP, exceto no protocolo CoAP. Já no protocolo MQTT-SN, a comunicação ocorre parcialmente através de ambos os protocolos UDP e TCP. O paradigma de comunicação original de cada um dos protocolos é exibido, contudo, cabe ressaltar que com as devidas extensões ainda em fase de

desenvolvimento, o CoAP suportaria também o modo P/A. Por fim, recomenda-se o uso do protocolo MQTT em aplicações que devem priorizar QoS, devido as características providas por seus três níveis, conforme apresentado anteriormente. Em termos de eficiência energética e QoS em simultâneo, o protocolo mais recomendado é o MQTT-SN.

VI. Desafios de Pesquisa e Tendências Futuras

A larga implantação das aplicações baseadas na Internet das Coisas depende de uma série de fatores que ainda requerem aprimoramentos a fim de alcançar níveis aceitáveis de segurança, interoperabilidade, desempenho e eficiência energética. Portanto, a seguir serão enumerados os principais desafios de pesquisa na área:

Segurança Leve: como discutido anteriormente, as especificações dos protocolos MQTT, MQTT-SN e CoAP assumem como premissa que tais protocolos farão o uso de protocolos existentes que exigem alto custo computacional. Desse modo, uma das principais limitações de segurança presente nesses protocolos consiste na carência de soluções apropriadas para o estabelecimento da confidencialidade, autenticidade, integridade, disponibilidade distribuição e gerenciamento de chaves de maneira viável [7] [8] [9] [30].

Disponibilidade: apesar das vantagens obtidas pela centralização da comunicação no componente *broker*, os protocolos MQTT e MQTT-SN estão susceptíveis ao problema da existência desse ponto único de falhas. Portanto, há a necessidade da exploração de alternativas que mitiguem a indisponibilidade em casos onde aconteçam falhas ou ataques que interrompam os serviços oferecidos pelo *broker*. O mesmo ocorre para o CoAP, que possui um servidor centralizado. Desse modo, estratégias calcadas na redundância podem ser usadas para mitigar esse problema, onde *brokers* e servidores são replicados para substituição em casos de falhas [31]. A extensão do CoAP [16] que visa permitir comunicação P/A sem *broker* pode ser outra alternativa para evitar pontos únicos de falhas. Contudo, essa abordagem pode impactar nas demais propriedades, conforme será discutido a seguir.

Eficiência Energética: uma das vantagens energéticas dos protocolos MQTT e MQTT-SN se dá por conta do paradigma de comunicação adotado, que possibilita a comunicação de um dispositivo para muitos. Além disso, para que seja possível prolongar ainda mais a vida útil das baterias desses dispositivos, técnicas de *duty-cycle* podem ser exploradas. Assim, com o desligamento temporário das interfaces de rádio dos dispositivos, as mensagens podem ser mantidas no *broker* até que a entrega seja possível [15] [17]. Contudo, o tempo excessivo de indisponibilidade de um dispositivo pode ocasionar sobrecargas de armazenamento no *broker*. Portanto, o gerenciamento eficiente do uso dessa prática permanece como uma questão em aberto. O MQTT-SN, especialmente, inclui a funcionalidade de gerenciamento de conexões dos dispositivos no *gateway*. Com isso, o tempo de restabelecimento de conexão diminui, tornando-se mais viável o desligamento temporário de suas interfaces de rádio [17]. O CoAP apresenta uma característica nativa que traz economia de energia que consiste no uso do protocolo UDP. Por outro lado, o modelo de comunicação utilizado (C/S) consome mais energia comparado ao paradigma adotado pelo MQTT (P/A). Desse modo, é possível que o futuro "CoAP P/A" possa apresentar resultados competitivos de desempenho. Contudo, a operação no modo Publicador/Assinante sem um *broker* (ou *gateway*) centralizado move a complexidade de processamento para os dispositivos (publicadores ou assinantes), podendo afetar a eficiência energética dos mesmos (*e.g.*, como a gerência de conexões, segurança e QoS) [6].

Escalabilidade: atualmente, experimentos comparativos demonstram que o CoAP possui problemas de escalabilidade, sendo incapaz de processar mais de mil requisições por segundo [2]. O protocolo MQTT, embora mais escalável que o CoAP, pode sofrer problemas de sobrecarga no *broker*. Portanto, a fim de tornar o sistema mais tolerante a falhas, técnicas de replicação do *broker* são uma potencial forma trazer escalabilidade ao MQTT. Novamente, o futuro modo P/A do CoAP pode ser uma alternativa. Porém, de acordo com o que foi discutido na Seção III, as abordagens distribuídas podem ser ineficientes no ponto de vista de escalabilidade. Assim, o modo *brokerless* do CoAP apresenta tal limitação.

VII. Conclusão

A Internet das coisas consiste em um paradigma promissor para aplicações em diversas áreas, oferecendo conectividade aos objetos do dia-a-dia. Os principais desafios existentes na IoT consistem na comunicação segura e eficiente entre dispositivos que apresentam restrições computacionais e energéticas, além da ciência de heterogeneidade na proposta de tecnologias e mecanismos. Na camada de aplicação, em especial, existem diferentes protocolos, tais como MQTT e CoAP. Ambos os protocolos são interessantes para o uso em aplicações da IoT, contudo, o protocolo MQTT e sua variação MQTT-SN têm sido apresentados como uma solução proeminente.

Em particular, para aplicações envolvendo sensores com capacidade energética restrita, o protocolo MQTT-SN é o mais adequado. Ademais, considerando cenários onde a confiabilidade não é um requisito crítico, o nível 0 de QoS do MQTT apresenta menor atraso do que o CoAP, conforme resultados obtidos e apresentados na Fig. 5. No entanto, a principal diferença conceitual desses protocolos está no paradigma de comunicação: enquanto o CoAP é indicado para cenários de comunicação *um-para-um*, o MQTT é ideal para cenários onde a comunicação deve partir de *muitos-para-muitos* dispositivos.

Por outro lado, as extensões que estão sendo desenvolvidas para o protocolo CoAP podem torna-lo bastante atraente para uso em aplicações diferentes daquelas atendidas em seu propósito original. É importante ressaltar que ambos os protocolos requerem melhorias como a implementação de mecanismos leves de segurança e energeticamente eficientes são necessárias a fim de superar as limitações para o emprego desses protocolos na IoT.

As principais contribuições deste trabalho consistem na revisão dos fundamentos, apresentação de ferramentas, discussão e direções futuras relacionadas aos protocolos MQTT e MQTT-SN, comparados a protocolos alternativos projetados para a camada de aplicação da IoT.

REFERÊNCIAS

[1] B. P. Santos, L. Silva, C. Celes, J. B. Borges, B. S. P. Neto, M. A. M. Vieira, L. F. M. Vieira, O. N. Goussevskaia, and A. Loureiro, "Internet das coisas: da teoria à prática," *Livro de Minicursos do XXXIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos (SBRC)*, vol. 1, pp. 15–52, 2016.

[2] A. Talaminos-Barroso, M. A. Estudillo-Valderrama, L. M. Roa, J. Reina-Tosina, and F. Ortega-Ruiz, "A Machine-to-Machine protocol benchmark for eHealth applications–Use case: Respiratory rehabilitation," *Comp. methods and prog. in biomedicine*, vol. 129, pp. 1–11, 2016.

[3] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[4] "RFC 7252: The Constrained Application Protocol (CoAP)," Disponível em: https://tools.ietf.org/html/rfc7252., 2018, acessadi em: 09/09/2018.

[5] OASIS, "MQTT Version 5.0 OASIS Standard," Disponível em: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html, 2019, acessado em: Março/2019.

[6] A. Stanford-Clark and H. L. Truong, "MQTT for sensor networks (MQTT-SN) protocol specification," *International business machines (IBM) Corporation version*, vol. 1, 2013.

[7] M. Singh, M. A. Rajan, V. L. Shivraj, and P. Balamuralidhar, "Secure MQTT for Internet of Things (IoT)," in *5th International Conference on Comm. Systems and Network Technologies*. IEEE, 2015, pp. 746–751.

[8] S. Andy, B. Rahardjo, and B. Hanindhito, "Attack scenarios and security analysis of MQTT communication protocol in IoT system," in *4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*. IEEE, 2017, pp. 1–6.

[9] D.-H. Mun, M. Le Dinh, and Y.-W. Kwon, "An assessment of internet of things protocols for resource-constrained applications," in *40th Annual Computer Software and Applications Conference (COMPSAC)*, vol. 1. IEEE, 2016, pp. 555–560.

[10] Iglesias-Urkia, Markel and Orive, Adrián and Barcelo, Marc and Moran, Adrian and Bilbao, Josu and Urbieta, Aitor, "Towards a lightweight protocol for Industry 4.0: An implementation based benchmark," in *International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM)*. IEEE, 2017, pp. 1–6.

[11] Palattella, Maria Rita and Accettura, Nicola and Vilajosana, Xavier and Watteyne, Thomas and Grieco, Luigi Alfredo and Boggia, Gennaro and Dohler, Mischa, "Standardized protocol stack for the internet of (important) things," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1389–1406, 2012.

[12] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: Real-time intrusion detection in the Internet of Things," *Ad hoc networks*, vol. 11, no. 8, pp. 2661–2674, 2013.

[13] M. Ruiz, E. Alvarez, A. Serrano, and E. Garcia, "The convergence between wireless sensor networks and the Internet of Things; challenges and perspectives: A survey," *IEEE Latin America Transactions*, vol. 14, no. 10, pp. 4249–4254, 2016.

[14] M. B. Yassein, M. Q. Shatnawi *et al.*, "Application layer protocols for the Internet of Things: A survey," in *International Conference on Engineering & MIS (ICEMIS)*. IEEE, 2016, pp. 1–4.

[15] B. Schütz, J. Bauer, and N. Aschenbruck, "Improving Energy Efficiency of MQTT-SN in Lossy Environments Using Seed-Based Network Coding," in *42nd Conference on Local Computer Networks (LCN)*. IEEE, 2017, pp. 286–293.

[16] M. Koster, A. Keranen, and J. Jimenez, "Publish-subscribe broker for the constrained application protocol (CoAP)," Disponível em: https://tools.ietf.org/html/draft-ietf-core-coap-pubsub-05, 2018, acessado em: 09/09/2018.

[17] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, "MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks," in *3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08)*. IEEE, 2008, pp. 791–798.

[18] E. Foundation, "Eclipse Mosquitto," Disponível em: https://projects.eclipse.org/projects/technology.mosquitto, 2019, acessado em: 09/09/2018.

[19] "Eclipse Paho Project," Disponível em: https://www.eclipse.org/paho/, 2019, acessado em: 09/09/2018.

[20] HiveMQ, "HiveMQ - Enterprise ready MQTT to move your IoT data," Disponível em: https://www.hivemq.com/ services.html, 2019, acessado: em Maio/2019.

[21] "Java MQTT lightweight broker," Disponível em: https://github.com/andsel/moquette, 2019, acessado em: 09/09/2018.

[22] GitHub, "Java MQTT lightweight broker," Disponível em: https://github.com/eclipse/californium., 2019, acessado em: 09/09/2018.

[23] Google, "CoapBlaster6," Disponível em: https://github.com/google/coapblaster, June 2019, acessado em: Juneço/2019.

[24] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *29th annual IEEE international conference on local computer networks*. IEEE, 2004, pp. 455–462.

[25] GitHub, "Porte do MQTT-SN para o Contiki," Disponível em: https://github.com/aignacio/homestark_mqtt_6lowpan_port., 2019, acessado em: 09/09/2018.

[26] Shodan, "Shodan," Disponível em: www.shodan.io, 2019, acessado em: 09/09/2018.

[27] M. Collina, M. Bartolucci, A. Vanelli-Coralli, and G. E. Corazza, "Internet of Things application layer protocol analysis over error and delay prone links," in *7th Advanced Satellite Multimedia Systems Conference and the 13th Signal Processing for Space Communications Workshop (ASMS/SPSC)*. IEEE, 2014, pp. 398–404.

[28] D. Thangavel, X. Ma, A. Valera, H.-X. Tan, and C. K.-Y. Tan, "Performance evaluation of MQTT and CoAP via a common middleware," in *Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*. IEEE, 2014, pp. 1–6.

[29] N. De Caro, W. Colitti, K. Steenhaut, G. Mangino, and G. Reali, "Comparison of two lightweight protocols for smartphone-based sensing," in *20th Symposium on Communications and Vehicular Technology in the Benelux (SCVT)*. IEEE, 2013, pp. 1–6.

[30] A. Esfahani, G. Mantas, R. Matischek, F. B. Saghezchi, J. Rodriguez, A. Bicaku, S. Maksuti, M. G. Tauber, C. Schmittner, and J. Bastos, "A Lightweight Authentication Mechanism for M2M Communications in Industrial IoT Environment," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 288–296, 2019.

[31] S. Abdallah, E. Najjar, and A. Kayssi, "A Round Robin Load Balancing and Redundancy Protocol for Network Routers," in *11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2012, pp. 1741–1747.

Silvio Ereno Quincozes é formado em Engenheira de Software, pela Universidade Federal do Pampa (UNIPAMPA), mestre em Ciências da Computação, pela Universidade Federal de Santa Maria (UFSM) e Doutorando em Computação, na Universidade Federal Fluminense (UFF). Tem interesse nas linhas de pesquisas relacionadas a Segurança da Informação, Redes de Computadores, Internet das Coisas, Sistemas de Detecção de Intrusões, Mineração de dados e Engenharia de Software.

Emilio Roberto Reginaldo Tubino possui graduação em Ciência da Computação pela Universidade Federal do Pampa (UNIPAMPA). Recebeu prêmio pela autoria do melhor artigo no Workshop de Trabalhos de Iniciação Científica e de Graduação do Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais, em 2016. Possui interesse e atua nas áreas de segurança da informação, redes de computadores e Internet das Coisas.

Juliano Fontoura Kazienko é professor adjunto da Universidade Federal de Santa Maria (UFSM). Possui graduação em Ciência da Computação pela Universidade do Vale do Itajaí (UNIVALI), mestrado em Ciência da Computação pela Universidade Federal de Santa Catarina (UFSC) e doutorado em Computação pela Universidade Federal Fluminense (UFF). Tem interesse e atuação em pesquisa nas áreas de redes sem fio, segurança da informação e computação ubíqua e pervasiva.