



**UNIVERSIDADE FEDERAL DA FRONTEIRA SUL
CAMPUS DE CHAPECÓ
CURSO DE CIÊNCIA DA COMPUTAÇÃO**

WILLIAN BORDIGNON GENERO

**AVALIAÇÃO DOS PROTOCOLOS MQTT E MQTT-SN NO CONTEXTO DA
INTERNET DAS COISAS**

**CHAPECÓ
2022**

WILLIAN BORDIGNON GENERO

**AVALIAÇÃO DOS PROTOCOLOS MQTT E MQTT-SN NO CONTEXTO DA
INTERNET DAS COISAS**

Trabalho de conclusão de curso apresentado como requisito para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal da Fronteira Sul.
Orientador: Prof. Dr. Marco Aurélio Spohn

CHAPECÓ
2022

Genero, Willian Bordignon

Avaliação dos protocolos MQTT e MQTT-SN no contexto da Internet das Coisas / Willian Bordignon Genero. – 2022.

38 f.: il.

Orientador: Prof. Dr. Marco Aurélio Spohn.

Trabalho de conclusão de curso (graduação) – Universidade Federal da Fronteira Sul, curso de Ciência da Computação, Chapecó, SC, 2022.

1. Internet das Coisas. 2. *MQTT*. 3. *MQTT-SN*. I. Spohn, Prof. Dr. Marco Aurélio, orientador. II. Universidade Federal da Fronteira Sul. III. Título.

WILLIAN BORDIGNON GENERO

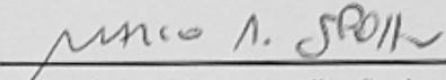
**AVALIAÇÃO DOS PROTOCOLOS MQTT E MQTT-SN NO CONTEXTO DA
INTERNET DAS COISAS**

Trabalho de conclusão de curso apresentado como requisito para obtenção do grau de Bacharel
em Ciência da Computação da Universidade Federal da Fronteira Sul.

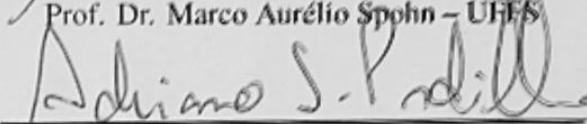
Orientador: Prof. Dr. Marco Aurélio Spohn

Aprovado em: 01/04/2022


BANCA AVALIADORA



Prof. Dr. Marco Aurélio Spohn - UFFS



Prof. Me. Adriano Sanick Padilha - UFFS



Prof. Dr. Bráulio Adriano de Mello - UFFS

RESUMO

A Internet das Coisas pretende revolucionar o mundo tecnológico atingindo áreas desde a saúde até os transportes, melhorando a qualidade de vida direta e indiretamente dos envolvidos. Os avanços nos dispositivos sem fios permitem a coleta de dados, processamento e atuações sem qualquer interação humana. Os desafios presentes nos dispositivos exigem pesquisas tanto em ambiente acadêmico quanto industrial para propor novas soluções. Apesar de diversos protocolos propostos, estes ainda carecem de avaliações de terceiros em seu desempenho para comprovar as excelências e deficiências. Para isso, foi implementado um ambiente de estresse ao nosso *broker* usando hospedagem em nuvem. Partindo de cenários com poucas necessidades de recursos até chegar a um ponto de claro gargalo. Os parâmetros de entrada alterados são quantidade de clientes e frequência de mensagem, como saída obtemos métricas de latência e perda de pacotes. Por fim, é analisado o comportamento dos protocolos MQTT e MQTT-SN.

Palavras-chave: Internet das Coisas. *MQTT*. *MQTT-SN*.

ABSTRACT

The Internet of Things intends to revolutionize the technological world, reaching areas from health to transport, improving the quality of life directly and indirectly for those involved. Advances in wireless devices allow data collection, processing and performances without any human interaction. The challenges present in the devices require research in both academic and industrial environments to propose solutions. Despite several proposed protocols, these still lack third-party evaluations of their performance to prove their excellences and deficiencies. For this, a stress environment was implemented for our broker using cloud hosting. Starting from scenarios with few resource needs until reaching a clear bottleneck point. The input parameters changed are number of clients and message frequency, as output we get latency and packet loss metrics. Finally, the behavior of the MQTT and MQTT-SN protocols is analyzed.

Keywords: Internet of Things. *MQTT*. *MQTT-SN*.

LISTA DE ILUSTRAÇÕES

Figura 1 – Funcionamento do protocolo MQTT (Fonte: Quincozes (22)).	13
Figura 2 – Formato de uma mensagem MQTT (Fonte: Al-Fuqaha (12)).	14
Figura 3 – Funcionamento do protocolo MQTT-SN (Fonte: Quincozes (22)).	15
Figura 4 – Configuração da arquitetura	22
Figura 5 – Resultados para o cenário 1	25
Figura 6 – Resultados para o cenário 2	25
Figura 7 – Resultados para o cenário 3	26
Figura 8 – Resultados para o cenário 4	26
Figura 9 – Resultados para o cenário 5	27
Figura 10 – Resultados para o cenário 6	28
Figura 11 – Resultados para o cenário 7	28
Figura 12 – Resultados para o cenário 8	29
Figura 13 – Resultados para o cenário 9	29
Figura 14 – Resultados para o cenário 10	30
Figura 15 – Resultados para o cenário 11	30
Figura 16 – Resultados para o cenário 12	31
Figura 17 – Resultados para o cenário 13	32
Figura 18 – Resultados para o cenário 14	33

LISTA DE TABELAS

Tabela 1 – Tabela descrevendo os cenários a serem testados.	23
---	----

SUMÁRIO

1	INTRODUÇÃO	10
1.1	OBJETIVOS	11
1.1.1	Objetivo Geral	11
1.1.2	Objetivos Específicos	11
1.2	JUSTIFICATIVA	12
2	REVISÃO BIBLIOGRÁFICA	13
2.1	PROTOCOLO <i>MQTT</i>	13
2.2	PROTOCOLO <i>MQTT-SN</i>	14
2.3	PROTOCOLO <i>COAP</i>	15
2.4	PROTOCOLO <i>DDS</i>	16
2.5	PROTOCOLO <i>AMQP</i>	16
2.6	PROTOCOLO <i>XMPP</i>	17
2.7	TRABALHOS RELACIONADOS	17
3	METODOLOGIA	21
3.1	INSTRUMENTOS E FERRAMENTAS DE DESENVOLVIMENTO	21
3.1.1	Brokers	21
3.1.2	Clientes	21
3.1.3	Hospedagem	21
3.1.4	Configurações de rede	22
3.2	CENÁRIOS DE AVALIAÇÃO	23
4	RESULTADOS	25
4.1	CENÁRIOS DE AVALIAÇÃO	25
4.1.1	Cenário 1 (Sub: 1, Pub: 1)	25
4.1.2	Cenário 2 (Sub: 1, Pub: 2)	25
4.1.3	Cenário 3 (Sub: 1, Pub: 4)	26
4.1.4	Cenário 4 (Sub: 1, Pub: 8)	26
4.1.5	Cenário 5 (Sub: 1, Pub: 16)	27
4.1.6	Cenário 6 (Sub: 2, Pub: 1)	28
4.1.7	Cenário 7 (Sub: 4, Pub: 1)	28
4.1.8	Cenário 8 (Sub: 8, Pub: 1)	29
4.1.9	Cenário 9 (Sub: 16, Pub: 1)	29
4.1.10	Cenário 10 (Sub: 2, Pub: 2)	30
4.1.11	Cenário 11 (Sub: 4, Pub: 4)	30
4.1.12	Cenário 12 (Sub: 8, Pub: 8)	31
4.1.13	Cenário 13 (Sub: 16, Pub: 16)	32
4.1.14	Cenário 14 (Sub: 16, Pub: 16, AWS)	33
4.2	ANÁLISE DOS RESULTADOS	33

5	CONCLUSÃO	35
5.1	TRABALHOS FUTUROS	35
	REFERÊNCIAS	36

1 INTRODUÇÃO

Considerada como parte da Internet do futuro, a Internet das Coisas (IoT) só é possível dado o crescente número de dispositivos conectados entre si. Segundo Al-Fuqaha et al. (12), IoT pode ser definida por “objetos físicos que veem, ouvem, pensam e desempenham papéis tendo que comunicar entre si, para compartilhar informações e coordenar decisões”, sendo essa a diferença da internet tradicional, pois os dispositivos inteligentes não precisam de interação humana para realizar suas ações.

Os objetos consistem em simples aparelhos equipados com um ou mais sensores, processador, memória, fonte de energia, rádio transmissor e opcionalmente um atuador. Esses dispositivos coletam informações sobre variáveis mecânicas, térmicas, químicas, ópticas, magnéticas, entre outras, podendo abranger todas as métricas presentes no ambiente. Tal avanço só foi possível dada as melhorias em diversas áreas como microeletrônica, comunicação e sensoriamento. Previsões geradas pela *Mordor Intelligence* esperam que o mercado de IoT gere 1,256 trilhão de dólar em 2025 (14), já a *Verified Market Research* especula US\$ 1,319 trilhão em 2026 (17). Esse novo mercado é capaz de incentivar outros campos da computação, como o de banco de dados, redes de computadores, *hardware* em geral e todo o ecossistema por volta disso.

Em pouco tempo novos locais de aplicação foram sendo propostos como, por exemplo, na área da saúde. Usando sensores é possível monitorar temperatura, pressão arterial, batimentos cardíacos dos pacientes à distância, permitindo que os mesmos permaneçam em casa durante o tratamento, diminuindo o número de leitos utilizadas nos hospitais. Tanto para pessoas em observação quanto em recuperação, caso alguma métrica altere esta será automaticamente detectada e a anomalia e as devidas providências são tomadas. Os cenários não se limitam a saúde: soluções são apresentadas para a agricultura e pecuária, transportes, indústrias, meio ambiente, segurança, educação, entre muitos outros.

Os bilhões de aparelhos geram um tráfego intenso de dados nas redes, necessitando de pesquisas para otimizar essa transferência. Diferentemente da Internet tradicional, na IoT ainda não foi determinado nenhum padrão de arquitetura; porém, o modelo de cinco camadas continua sendo usado. Dentre todas as camadas a de aplicação é a que mais recebe atenção da comunidade. O crescente número de dispositivos conectados e de cenários aplicáveis, cada qual com suas particularidades, traz consigo a necessidade de soluções para resolver os problemas consequentes. Como a maior parte dessa rede é ou será de sensores sem fios que tendem a ter tamanho reduzido e por consequência seus recursos também, dificultam as pesquisas. Chang; Srirama; Buyya (4) listam os principais desafios sendo: largura de banda, latência, ininterrupção, processamento, energia e segurança.

Inúmeros protocolos estão sendo propostos ou até mesmo adaptações dos já utilizados anteriormente. Um desses protocolos propostos é o *Message Queuing Telemetry Protocol* (MQTT), iniciado nos laboratórios da *International Business Machines Corporation* (IBM) uti-

liza do paradigma de comunicação *Publish/Subscribe* para oferecer um consumo de recursos reduzido. No mesmo laboratório foi proposto uma adaptação do MQTT focado em dispositivos sem fios (MQTT-SN) extremamente limitados de recursos (processamento, memória e principalmente de energia).

Dispositivos sem fios geram desafios para as devidas implementações, além do que os diferentes ambientes de aplicação resultam em diferentes requisitos. Por exemplo, em cenários de saúde é solicitado confiabilidade, velocidade, interoperabilidade e tratamento de falhas; por outro lado, uma aplicação de agricultura ou de monitoramento de clima em locais inóspitos, como florestas, exige longa duração de bateria.

A bateria é um dos mais importantes recursos para sensores sem fios. Frequentemente esses sensores possuem uma estrutura mínima para coletar amostras e enviar para um servidor processá-las ou receber alguma instrução solicitando certa ação. A duração da bateria não só é importante para cenários onde é difícil chegar até o dispositivo para substituição da bateria, mas também quando a manutenção é cara financeiramente ou com manuseio inviável. Aplicações onde a frequência de mensagens é alta, ou são enviados grandes pacotes, afetam a vida útil do sensor por manter o rádio transmissor ativo. Um dispositivo sem energia não é capaz de executar suas tarefas e, qualquer que seja a aplicação, não é o comportamento desejado.

Outro desafio é tratar a heterogeneidade dos objetos que são diferentes em implementação e recursos. Surgem questionamentos de como garantir endereçamento único para tantos aparelhos ou de como prover roteamento eficaz para tantos dados. A segurança precisa ser adaptada, pois os métodos atuais requerem muitos recursos; por outro lado, com fraca criptografia as mensagens recebidas podem ser alteradas afetando perigosamente a confiabilidade do conteúdo.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Avaliar os protocolos de comunicação MQTT e MQTT-SN do tipo publicador/assinante no contexto da Internet das Coisas.

1.1.2 Objetivos Específicos

- Buscar junto a literatura outros trabalhos de avaliação.
- Executar testes a partir de cenários definidos.
- Realizar um comparativo entre os resultados obtidos e os trabalhos relacionados.

1.2 JUSTIFICATIVA

É inegável a expansão da Internet das Coisas. Com frequência são publicados novos protocolos, adaptações ou melhorias. A participação tanto da comunidade científica quanto da industrial gera expectativa em torno do novo paradigma, ao mesmo tempo o avanço é notável. Eventuais melhorias dos protocolos e *hardwares* envolvidos encaminhará a tecnologia para um nível de plena adoção pela comunidade.

Um estudo feito pela Gartner (13) mostra o estado atual de adoção pela comunidade. Já superando o pico de expectativas gerada principalmente por publicidades exibindo resultados de sucesso. A atual posição é delicada pois muitos experimentos falharam, reduzindo significativamente os investimentos e aumentando a pressão sobre os desenvolvedores e pesquisadores. A perspectiva é que leve entre 2 a 5 anos para a IoT ser adotada no mercado.

Avaliar um protocolo proposto por terceiros tem muitos pontos positivos, um deles é detectar as qualidades e fraquezas do mesmo. Conhecer profundamente as características ajuda os desenvolvedores e profissionais da área a escolher com efetividade entre os inúmeros protocolos qual deles melhor atende aos requisitos da sua aplicação.

A comunidade científica já realizou trabalhos com este propósito antes, mas a quantidade de cenários é incontável. Desse modo, sempre há métricas, número de objetos, distância entre eles, até mesmo o objetivo que podem ser alterados, levando a um resultado totalmente diferente de outros estudos. Neste âmbito, uma opção também é validar as análises de terceiros, para garantir a qualidade da pesquisa.

A melhoria na qualidade dos serviços prestados em ambientes que fazem uso de sensores sem fios será notável com o surgimento de novos produtos. Com tantos setores vindo a serem automatizados, a qualidade de vida das pessoas irá melhorar, aperfeiçoando a eficiência e produtividade, prevenindo doenças, entre muitos outros benefícios. O monitoramento de atividades com potencial risco humano podem vir a serem substituídas por sensores que em tempo real fazem a coleta de amostras como, por exemplo, fiscalizar vazamentos e pressão em dutos de gás e óleo. O retorno financeiro também é um atrativo para impulsionar pesquisas e novas soluções.

2 REVISÃO BIBLIOGRÁFICA

2.1 PROTOCOLO MQTT

Desenvolvido por Andy Stanford-Clark e Arlen Nipper em 1999, MQTT é um protocolo para camada de aplicação utilizado na comunicação de *machine-to-machine* (M2M), *server-to-server* (S2S) e *machine-to-server* (M2S) na Internet das Coisas.

Foi padronizado pela *Organization for the Advancement of Structured Information Standards* (OASIS) em 2013. A utilização em aplicações por grandes empresas, como Facebook, e o suporte às plataformas de *cloud computing* Amazon AWS, Google Cloud, IBM Cloud e Microsoft Azure alavancaram a expansão do protocolo tornando-o um dos mais utilizados no meio.

Diferentemente da Internet tradicional que utiliza o paradigma de comunicação *Client-Server*, o MQTT faz uso do paradigma *Publish/Subscribe* (18): os clientes não fazem solicitações para receber dados e também não guardam informações sobre os destinos das mensagens, tornando o *broker* responsável pelos encaminhamentos.

O MQTT é formado por três componentes: *subscriber*, *publisher* e o *broker*. Os clientes se inscrevem (*subscribers*) em determinado tópico de um *broker* para receber os dados de interesse que serão enviados por publicadores (*publishers*) até o *broker*. Esse formato de comunicação pode comportar os mecanismos de roteamento *one-to-one*, *one-to-many* e *many-to-many*(29).

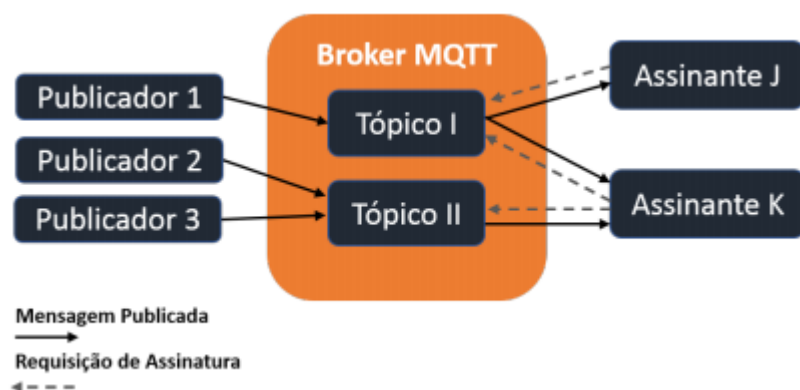


Figura 1 – Funcionamento do protocolo MQTT (Fonte: Quincozes (22)).

Como pode ser visto na figura 1, o assinante J possui interesse no tópico I e o assinante K tem interesse nos tópicos I e II. Portanto, quando uma mensagem for enviada pelo publicador 1 até o *broker*, será feita uma verificação de interessados do tópico I e o servidor encaminhará a mensagem para os assinantes J e K.

Assim como o *Hypertext Transfer Protocol* (HTTP), o MQTT é construído sobre o protocolo de transporte *Transmission Control Protocol* (TCP), mas focado em dispositivos

limitados de recursos, com enlaces com pequena largura de banda, alta latência e/ou não confiáveis (11) e com restrição energética (18) e necessidade de escalabilidade (22).

0	1	2	3	4	5	6	7
Message Type				UDP	QoS Level		Retain
Remaining Length (1~4 bytes)							
Variable Length Header (Optional)							
Variable Length Message Payload (Optional)							

Figura 2 – Formato de uma mensagem MQTT (Fonte: Al-Fuqaha (12)).

Uma mensagem MQTT possui um cabeçalho fixo de 2 bytes contendo o tipo de pacote (conexão, assinatura, publicação, entre outros 12 tipos), *flags* (somente em mensagens tipo *PUBLISH* para verificar duplicações, qualidade de serviço e retenção no *broker*) e o tamanho restante da mensagem. Um cabeçalho opcional pode ser adicionado, assim como a carga útil da mensagem com um tamanho máximo de 65.535 bytes no formato UTF-8 (3).

O MQTT tem suporte a três níveis de qualidade de serviço, sendo eles:

- 0 - No máximo uma vez: a mensagem é entregue uma vez ou sequer chega ao destino;
- 1 - Pelo menos uma vez: a mensagem é reenviada até receber uma confirmação, mesmo que ocasione redundância;
- 2 - Exatamente uma vez: sempre entrega uma vez, tornando-se o modo mais confiável apesar de mais lento.

2.2 PROTOCOLO MQTT-SN

Extensão do MQTT para redes de sensores, o MQTT-SN (*Sensor Networks*) foi desenvolvido para dispositivos extremamente limitados computacionalmente, em armazenamento e em capacidade energética. Por serem utilizados em locais remotos ou de difícil acesso e/ou manuseio, os dispositivos sem fios possuem problemas quando precisam enviar excesso de meta-dados ou serem requisitados muitas vezes, pois diminui a vida útil da bateria para transferência de dados não essenciais. Tais dificuldades são solucionadas utilizando paradigma *Publish/Subscriber* por conter um *broker* responsável em manter o endereço dos sensores e encaminhar as mensagens, mantendo baixa complexidade para os clientes. Outra funcionalidade do dispositivo é ficar ocioso por longos períodos de tempo para estender a vida útil da bateria. Por ser uma extensão do MQTT, tenta-se o máximo possível se assemelhar com o mesmo, resultando em pequenas modificações para fazer a integração de ambos (16).

O tamanho de uma mensagem MQTT-SN é reduzido por ser baseada em protocolos das camadas inferiores que oferecem pequena largura de banda como, por exemplo, o padrão IEEE

802.15.4 que provê uma largura de banda máxima de 250 kbit/s. Para ser resistente a erros de transmissão o pacote contém um tamanho de 128 bytes e metade deles são utilizados pela camada de enlace, rede, transporte e para segurança (25).

Como pode ser visto na figura 3, além dos já conhecidos publicadores, assinantes e o *broker*, o MQTT-SN utiliza uma estrutura adicional chamada de *gateway* que realiza a interoperabilidade entre os dispositivos internos e externos da rede de sensores (22). O *gateway* pode ser configurado em dois formatos: “transparente”, onde para cada cliente MQTT-SN será configurada e mantida uma conexão com o *broker* tornando fácil de implementar; e o *gateway* “agregado”, onde somente uma conexão será mantida com o *broker*, aumentando a complexidade de implementação e melhorando em termos de escalabilidade.

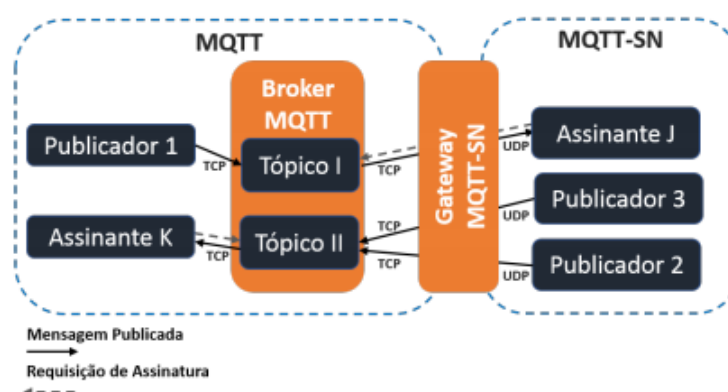


Figura 3 – Funcionamento do protocolo MQTT-SN (Fonte: Quincozes (22)).

Outro diferencial é o uso de *User Datagram Protocol* (UDP) para camada de transporte ao invés de TCP, motivado pelo UDP ser sem conexão, tornando possível que o dispositivo desligue sua custosa interface de rede (24).

Para solucionar o pequeno tamanho de pacote disponível, diversas soluções foram abordadas como, por exemplo: fragmentar e montar as mensagens, o que não se mostrou eficiente; dividir a mensagem para cada campo, mas em mensagens do tipo *PUBLISH* o nome do tópico continua ocupando grande parte do pacote. Como resultado, o método efetivamente utilizado é o de renomear o nome do tópico para um identificador de dois bytes, quando o mesmo for registrado (16).

Os níveis de qualidade de serviço permanecem os mesmos do MQTT com a adição do nível -1 onde torna desnecessário estabelecer conexão antes da troca de dados (28).

2.3 PROTOCOLO COAP

Constrained Application Protocol foi criado pelo grupo *Internet Engineering Task Force* (IETF) com a transferência de dados baseado no *Representational State Transfer* (REST) para facilitar a interoperabilidade com o HTTP e consumir serviços utilizando *Uniform Resource Identifiers* (URIs). Seu diferencial do HTTP é o uso do UDP ao invés do TCP (23), motivado

por remover a sobrecarga do TCP, reduzindo os requisitos de largura de banda. Outra distinção é o suporte de roteamento *unicast* e *multicast*.

Devido ao uso do UDP, algumas funcionalidades se fazem necessárias para garantir entrega confiável, como o mecanismo *Stop-and-Wait*, onde para cada mensagem enviada será aguardado um retorno confirmando a entrega. Além dessas, outros cinco tipos de mensagens podem ser enviados.

Em termos de segurança é discutido por Karagiannis et al. (18), pois o CoAP propõe *Datagram Transport Layer Security* (DTLS) para proteger as transações. Porém, DTLS não foi projetado para IoT, resultando em problemas como a falta de suporte para roteamento *multicast* e necessidade de *handshake*, ocasionando em pacotes adicionais trafegando na rede e levando a um maior consumo de energia e de largura de banda.

2.4 PROTOCOLO DDS

Data-Distribution Service provê comunicação *machine-to-machine* utilizando o paradigma *publish/subscribe*. Seu objetivo é facilitar a comunicação distribuída eficiente e foca principalmente em sistemas em tempo real (21).

O grande diferencial presente nesse protocolo é o suporte a 23 políticas de QoS, com critérios atingindo segurança, urgência, prioridade, durabilidade, confiabilidade, entre outros (12).

Possui descobrimento da topologia de rede de forma dinâmica, sem a necessidade de um *broker* como o MQTT e, desta forma, elimina um ponto único de falha (2).

Cinco componentes são necessários para um fluxo completo de dados usando o formato de entrega centrado de dados, sendo:

- *Publisher*: responsável pela distribuição de dados;
- *DataWriter*: atua entre a aplicação e o *publisher* analisando se a estrutura está de acordo com o esperado do tópico;
- *Subscriber*: disponibiliza os dados para as aplicações;
- *DataReader*: recebe os dados providos da rede;
- *Topic*: identificador para publicadores e inscritos encontrarem conteúdo em que ambos estão interessados.

2.5 PROTOCOLO AMQP

Advanced Message Queuing Protocol é um padrão aberto de comunicação orientado a mensagens, originalmente proposto para a indústria de sistemas financeiros, em 2006. Outro objetivo é a interoperabilidade entre sistemas heterogêneos. Dois tipos de mensagem podem ser enviadas: “crúas”, que são simples e com um tamanho menor; e, as “anotadas”, onde pode ser

adicionado um cabeçalho e informações sobre durabilidade, tempo de vida, prioridade, entre outras (12).

O funcionamento é semelhante ao MQTT, formado pelos já conhecidos *broker*, publicador e consumidor, com a adição de duas estruturas dentro do broker: uma fila de mensagens para armazenar e entregar em ordem sequencial; e, uma *exchange*, que é responsável por fazer o encaminhamento das mensagens para suas devidas filas (26).

Assim como o MQTT, o AMQP oferece confiabilidade na entrega de mensagens com três níveis de qualidade, sendo elas: ao menos uma, no mínimo uma ou exatamente uma (12).

2.6 PROTOCOLO XMPP

Desenvolvido pela comunidade Jabber para oferecer um protocolo aberto, seguro, descentralizado e livre de spam, o *Extensible Messaging and Presence Protocol* tornou-se padrão do grupo IETF e é utilizado para chat, voz, vídeo e telepresença entre múltiplas aplicações (12).

Segurança é oferecida usando criptografia SSL/TLS e autenticação com *Simple Authentication and Security Layer* (SASL). Apesar do uso do formato *Extensible Markup Language* (XML) fornecer uma boa interoperabilidade, o protocolo peca em aspectos essenciais para IoT como alto consumo de CPU, largura de banda e nenhum suporte a QoS (22).

2.7 TRABALHOS RELACIONADOS

Talaminos-Barroso et al. (27) utilizaram 41 máquinas para realizar um extenso *benchmark* com objetivo de avaliar o comportamento dos protocolos de aplicação baseados no paradigma de comunicação *publish/subscribe* em uma situação de *eHealth* para reabilitação de respiração. Tal reabilitação pode acontecer tanto no hospital quanto em casa, e consiste em atividades físicas aeróbicas com monitoramento de batimentos cardíacos, saturação de oxigênio do sangue e consumo de calorias, necessitando-se acompanhamento médico.

Para os protocolos DDS, MQTT, AMQP, XMPP, CoAP e *Java Message Service* (JMS), cinco métricas foram avaliadas sendo elas: uso de CPU, memória, consumo de largura de banda, latência e tremulação. As mensagens enviadas não tem retorno de confirmação.

O fato do DDS ser distribuído, resultou em um maior consumo de recursos conforme aumento de mensagens comparado aos demais. Por outro lado, apresentou o melhor resultado em latência e tremulação. O XMPP apresentou péssimos resultados em todas as métricas. AMQP, CoAP e MQTT tiveram desempenho semelhante entre si, pois necessitam de menos recursos.

Com o objetivo de facilitar desenvolvedores na escolha de protocolos para ambientes limitados de recursos, Mun; Le Dinh; Kwon (19) testaram o tempo de transmissão, eficiência energética e uso de CPU para os protocolos CoAP, MQTT e MQTT-SN. As mensagens, que podem ser de diferentes tamanhos, partindo de 128 até 1920 bytes, são enviadas para três

servidores (Local Utah-EUA, Amazon Web Service Oregon-EUA e Amazon Web Service Tóquio-Japão) que respondem com uma confirmação.

O tempo de transmissão é semelhante entre CoAP e MQTT, que diminui conforme aumenta o tamanho do pacote. O MQTT-SN, por sua vez, desempenha justamente ao contrário, aumentando o tempo quando se eleva o tamanho do pacote. Tal comportamento se repete no consumo de energia e de CPU, tornando-se o protocolo com os piores resultados. Mun; Le Dinh; Kwon (19) afirmam que o péssimo desempenho é devido à implementação deficiente da biblioteca Eclipse Paho MQTTSN.

Tanto para gasto de energia quanto de CPU, o CoAP é superior ao MQTT para pacotes menores que 1024 bytes, e inferior para pacotes maiores, pois o CoAP fragmenta suas mensagens maiores de 1kB.

Naik (20) publicou uma avaliação técnica envolvendo os protocolos para IoT AMQP, CoAP, MQTT e HTTP, estes escolhidos por serem aceitos e emergentes na comunidade. HTTP por não ser desenhado para as restrições da IoT, apresenta o maior tamanho de mensagem, consumo de processamento, largura de banda, latência e menor confiabilidade; por outro lado, tem superioridade em interoperabilidade, recursos adicionais e padronização.

No aspecto tamanho de mensagem e sobrecarga, o CoAP, apesar de ter um cabeçalho maior que o MQTT, possui uma mensagem menor por operar sobre UDP e aumentando para o AMQP pelo suporte a segurança, confiabilidade, entre outros. Fato este que ocasiona no maior gasto de processamento e largura de banda. CoAP e MQTT tem consumo semelhante de recursos.

O uso de TCP acarreta em maior latência de transmissão para o MQTT, AMQP e HTTP devido o estabelecimento de conexões, controle de congestionamento e de fluxo. Os três níveis de qualidade de serviço garantem ao MQTT maior confiabilidade entre os 4 protocolos, seguido pelo AMQP. MQTT detém poucas funcionalidades de segurança, diferentemente do AMQP.

Apesar dos contras, o MQTT é amplamente adotado mas ainda não é um padrão global, enquanto o AMQP é majoritariamente empregado em grandes projetos como o *NASA's Nebula Cloud Computing*. O CoAP, por sua vez, está emergindo e ganhando suporte em algumas plataformas, enquanto o HTTP é limitado ao fato do baixo desempenho e tamanho exagerado dos pacotes.

Motivados em melhorar a comunicação de aplicações robóticas que geralmente utilizam *Robot Operating System* baseado no lento e consumidor de recursos HTTP, Amaran et al. (1) comparam os protocolos CoAP e MQTT-SN por ambos usarem UDP como base. Para analisar o tempo de transmissão 10.000 mensagens foram enviadas entre dois dispositivos configurados em rede local.

A primeira mensagem transferida foi 59% mais rápida por conta do CoAP. Esse comportamento é explicado devido ao MQTT-SN necessitar registrar um tópico antes de compartilhar dados. Já para o tempo médio de transmissão, o MQTT-SN com 5,9 ms executou 30% mais rápido que o CoAP.

Aures; Lübben (2) avaliam tecnicamente DDS, MQTT e VSL (Virtual State Layer) do ponto de vista de desenvolvedores e apontam os pontos fortes e fracos de cada um deles. Apesar de ser superado pelo VSL, o já conhecido DDS apresenta robustez em integridade de dados, autenticação, controle de acesso e na estruturação com tratamento agnóstico de dados. Ambos oferecem criptografia e serialização. A qualidade de serviço, monitoramento e gerenciamento em tempo real é melhor por parte do DDS. Em contraponto, a sobrecarga dos dois é um ponto negativo, assim como a complexidade de uso do DDS. O MQTT, por sua vez, oferece simplicidade e baixa sobrecarga de pacote, mas ao custo de recursos fracos ou não oferecidos comparado com os demais protocolos.

Usando sensores para coletar dados de frequência cardíaca, oxigênio no sangue, temperatura, resistência e condutividade elétrica da pele, orientação e acelerômetro, Chen; Kunz (6) observaram a execução do CoAP, DDS, MQTT e uma versão autoral chamada *Custom UDP* num cenário de aplicação *eHealth*. O ambiente é configurado usando laptop, Arduino Uno e Raspberry Pi 2 conectados em uma rede simulada limitada e com baixa confiabilidade, sendo que para cada teste são transmitidos 981.600 bytes em um intervalo de 10 minutos.

O consumo de largura de banda para o CoAP e *Custom UDP* se mantém o mesmo independente da quantidade de pacotes perdidos ou da latência da rede justamente pelo uso de UDP como protocolo de transporte, diferentemente de DDS e MQTT que aumentam o consumo. A latência experimentada pelo MQTT tem uma taxa de crescimento muito alta à medida que a latência do sistema ou perda de pacotes aumentam. CoAP, DDS e *Custom UDP* apresentam um desempenho semelhante, quase não alterando a latência.

Os protocolos baseados em TCP realizam retransmissões para pacotes perdidos na rede. Tal comportamento não se repete para os protocolos baseados em UDP. Para a métrica de consumo de rede, a cada 100 bytes de *payload* outros 289 bytes são usados para controle no DDS, 76 no MQTT, CoAP com 36 e UDP 32. Em compensação, em termos de velocidade, o DDS apresenta aproximadamente metade da latência do MQTT; porém, o CoAP e UDP continuam melhores nesse quesito.

Com o objetivo de auxiliar desenvolvedores na escolha de protocolos, Cosmi; Mota (7) mensuraram qualitativa e quantitativamente os protocolos AMQP, CoAP, MQTT e MQTT-SN. Na parte técnica foram examinados aspectos como restrições de processamento, qualidade de sinal de rede, suporte a grandes quantidade de dados, entre outros. Diferentemente dos demais o AMQP exige mais recursos para oferecer melhores funcionalidades. CoAP, MQTT e MQTT-SN possuem foco em dispositivos restritos, mas divergem nas características, podendo ser visto no suporte a envio de dados em tempo real, reconhecimento de outros dispositivos e confiabilidade de entrega.

Usando um ESP8266 para simular os dispositivos IoT, observou-se melhores resultados do MQTT no quesito de retransmissões para troca de mensagens em um intervalo de tempo menor que 500 ms; acima disso, por influência do algoritmo de Nagle, aumenta as retransmissões. O CoAP mostrou-se sensível para intervalos menores que 500 ms. De acordo com o crescimento

do *payload* da mensagem, o MQTT diminui a taxa de retransmissões, enquanto o CoAP mantém em média elevados 3%. Outra métrica é a análise de *Round Trip Time*, ou tempo de recebimento, em uma rede sem e com tráfego de fundo: tanto o CoAP como o MQTT mostraram um aumento considerável no tempo.

Chaudhary; Peddoju; Kadarla (5) em seu estudo buscaram entender o impacto da transmissão de dados entre dispositivos IoT e servidores externos utilizando redes cabeadas, sem fio, 2G, 3G e 4G para os protocolos CoAP, MQTT e AMQP. Foi observado que o número de pacotes gerados para entregar uma mensagem aumenta significativamente quando é requisitado ao MQTT nível QoS 1 ou 2, o mesmo se repetindo para a rede 3G com perdas de dados. A vazão de poucas mensagens é similar entre os protocolos, mas para grande volumes de dados a qualidade de serviço do MQTT garante maior vazão devido possíveis perdas de pacotes e as retransmissões consequentes. Tais fatos replicam-se no consumo de largura de banda, pois para retransmitir mensagens mais pacotes serão enviados pela rede.

Como pode ser visto, muitos outros trabalhos já buscaram realizar a avaliação dos protocolos baseados no paradigma publicador/assinante. Mesmo assim, ainda há uma ampla variedade de cenários que podem ser testados. Este trabalho busca preencher a lacuna presente nos trabalhos acadêmicos para avaliação em termos de transmissão de dados ajustando a intensidade de dados trafegados.

3 METODOLOGIA

O presente trabalho busca de forma experimental analisar o desempenho na transmissão de dados dos protocolos MQTT e MQTT-SN no contexto de Internet das Coisas. A análise de dados, do tipo quantitativa, será realizada através de um estudo sobre as métricas de latência e perda de pacotes. As etapas para a execução da pesquisa encontram-se descritas a seguir.

3.1 INSTRUMENTOS E FERRAMENTAS DE DESENVOLVIMENTO

3.1.1 Brokers

Mosquitto (8) é uma implementação de código aberto de servidor para mensagens MQTT que suporta as versões 3.1, 3.1.1 e 5.0. O projeto faz parte da *Eclipse Foundation* e está disponível para uma grande gama de dispositivos e sistemas operacionais. Será usado na sua versão 1.6.9 como broker na comunicação MQTT.

Dado que o **Mosquitto** não suporta comunicação usando MQTT-SN, o **RSMB** (10) surge como solução e usa a mesma base de código. Essa implementação também faz parte da comunidade *Eclipse*. O mesmo projeto fornece o *gateway* necessário para converter mensagens MQTT-SN em MQTT. Será usado como broker na comunicação MQTT-SN usando a versão com *git commit id 36fd4ba*.

3.1.2 Clientes

A comunidade MQTT oferece vários projetos para utilizar como clientes. No presente trabalho será usado uma implementação de Eclipse Paho MQTT (9). A biblioteca fornece métodos para instanciar inscritos e publicadores.

Por outro lado, MQTT-SN tem poucas implementações disponíveis para serem usadas. A comunicação será feita usando a implementação de Humfrey (15). Algumas alterações foram necessárias no projeto inicial, principalmente sobre o *payload* da mensagem. Essas mudanças são permitidas pela licença MIT.

Para ambos os casos foi buscado o máximo de automatização, implementamos alguns *scripts* para inicializar um processo para cada cliente. Da mesma forma, para converter as mensagens transferidas nas métricas desejadas pelo trabalho.

3.1.3 Hospedagem

Tanto o broker **Mosquitto** quanto o **RSMB** serão hospedados na nuvem, mais especificamente em Virgínia do Norte - EUA. O serviço usado é o Amazon Web Service EC2 e a máquina dispõe de 1 núcleo Intel Xeon, 1 GiB de memória RAM e executa o sistema operacional



Figura 4 – Configuração da arquitetura

Ubuntu 20.4. Adotamos o uso de AWS pelo conhecimento prévio do autor nos serviços e por ser de acesso gratuito, facilitando que terceiros possam replicar a avaliação usando a mesma configuração.

Os clientes, publicadores e assinantes, serão mantidos em máquina própria localizada em Santa Catarina - Brasil. A máquina é configurada por 8 núcleos Apple M1, 8 GB de memória RAM com sistema operacional macOS Monterey 12.1.

Na figura 4 temos uma noção de como nossa estrutura está disposta geograficamente. O enlace é meramente ilustrativo, pois não temos informações sobre a rede.

3.1.4 Configurações de rede

As características do enlace de saída da máquina local consistem em comunicação Wi-Fi 5Ghz entre máquina e roteador, e via fibra ótica até o provedor. A velocidade de conexão contratada é de 200 Mb/s para *download* e 100 Mb/s para *upload*. A AWS não fornece informações numéricas sobre seu enlace, apenas que é de baixa performance.

Para confirmação foram executados testes de velocidade para diferentes servidores que

resultou em uma taxa efetiva média de *download* de 58,78 Mbps e 88,6 Mbps para *upload*.

Outra informação importante sobre a rede é o *Round Trip Time* da máquina local até nossa máquina AWS EC2. Nos testes realizados o tempo médio foi de 147,677 ms.

3.2 CENÁRIOS DE AVALIAÇÃO

Os cenários consistem em testes que avaliarão o desempenho dos protocolos MQTT e MQTT-SN nas métricas de latência e perdas de pacotes utilizando um ambiente real entre máquina local e nuvem. Um número *n* de clientes serão dispostos conectados em uma rede. Esses dispositivos terão as funções divididas de acordo com o paradigma publicador/assinante.

Os parâmetros a serem modificados para coletar dados serão a quantidade de mensagens enviadas, o número de publicadores e de assinantes. Dessa forma, será possível avaliar o comportamento dos protocolos sob diferentes cenários com maior ou menor intensidade de dados sendo transferidos.

Cenário	Nº assinantes	Nº publicadores	Nº msg/s
1	1	1	(1, 10, 100, 200, 400, 700, 1000)
2	1	2	(1, 10, 100, 200, 400, 700, 1000)
3	1	4	(1, 10, 100, 200, 400, 700, 1000)
4	1	8	(1, 10, 100, 200, 400, 700, 1000)
5	1	16	(1, 10, 100, 200, 400, 700, 1000)
6	2	1	(1, 10, 100, 200, 400, 700, 1000)
7	4	1	(1, 10, 100, 200, 400, 700, 1000)
8	8	1	(1, 10, 100, 200, 400, 700, 1000)
9	16	1	(1, 10, 100, 200, 400, 700, 1000)
10	2	2	(1, 10, 100, 200, 400, 700, 1000)
11	4	4	(1, 10, 100, 200, 400, 700, 1000)
12	8	8	(1, 10, 100, 200, 400, 700, 1000)
13	16	16	(1, 10, 100, 200, 400, 700, 1000)
14 ¹	16	16	(1, 10, 100, 200, 400, 700, 1000)

Tabela 1 – Tabela descrevendo os cenários a serem testados.

Na tabela 1 estão descritos todos os parâmetros que devem ser utilizados para cada cenário. Partindo de uma configuração que utiliza poucos recursos para uma que gera estresse em nosso broker. Os cenários foram definidos usando a cardinalidade 1:N e N:N.

O número máximo de 16 clientes foi definido após limitações de recursos na máquina local e dificuldade em sincronizar todos os processos.

Todas as mensagens terão 20 bytes, serão enviadas durante 10 segundos para um mesmo tópico e configuradas com QoS de nível 0. Mensagens que não chegarem ao assinante em um intervalo de 30 segundos serão consideradas perdidas. As amostras contarão com métricas de média e desvio padrão para latência e porcentagem de perda de pacotes.

¹ Clientes hospedados em máquina AWS

Os testes serão executados durante vários dias e em diferentes horários, além de que para cada conjunto de parâmetros serão feitas 10 execuções para evitar peculiaridades momentâneas do ambiente.

Por fim, a pesquisa contará com um comparativo entre os resultados obtidos e os presentes em trabalhos relacionados. Um deles é o de Mun; Le Dinh; Kwon (19) que como visto anteriormente analisaram os efeitos gerados pela mudança no tamanho dos pacotes enviados. Cosmi; Mota (7), por sua vez, coletaram resultados de retransmissões de pacotes ajustando o intervalo de tempo e tamanho do pacote da mensagem.

4 RESULTADOS

4.1 CENÁRIOS DE AVALIAÇÃO

4.1.1 Cenário 1 (Sub: 1, Pub: 1)

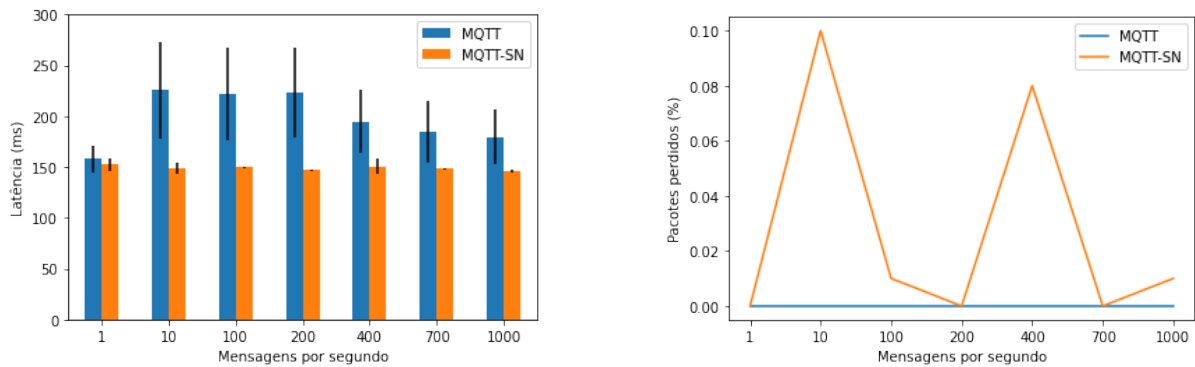


Figura 5 – Resultados para o cenário 1

Considerado o cenário que necessita da menor quantidade de recursos, apresentou resultados estáveis onde a latência média do MQTT-SN fica em aproximadamente 150 ms comparado com 200 ms do MQTT. O desvio padrão da latência por parte do MQTT foi consideravelmente superior. As perdas de pacotes foram mínimas com no máximo 0.1% para 10 msg/s no MQTT-SN.

4.1.2 Cenário 2 (Sub: 1, Pub: 2)

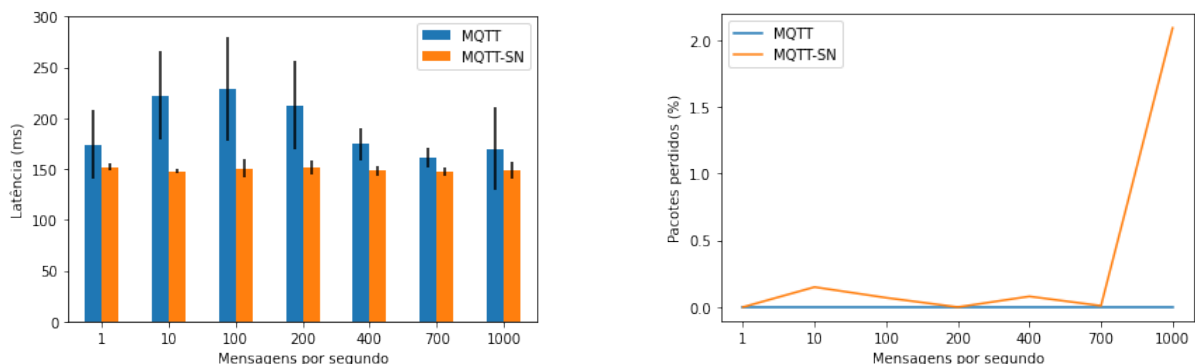


Figura 6 – Resultados para o cenário 2

Os resultados mantiveram-se semelhantes aos do cenário 1, porém com um leve aumento nos pacotes perdidos para o MQTT-SN durante a transmissão de 1000 msg/s.

4.1.3 Cenário 3 (Sub: 1, Pub: 4)

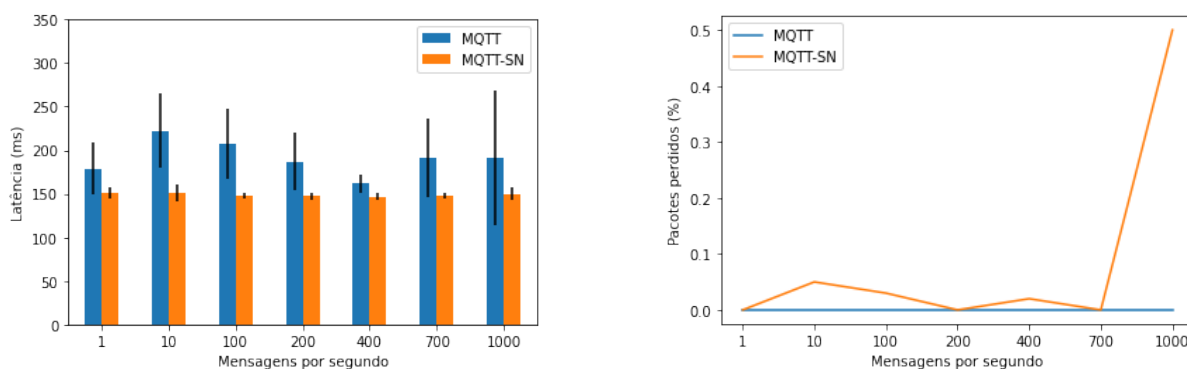


Figura 7 – Resultados para o cenário 3

Nesse cenário é notável o aumento da latência média do MQTT para 700 msg/s e 1000 msg/s.

4.1.4 Cenário 4 (Sub: 1, Pub: 8)

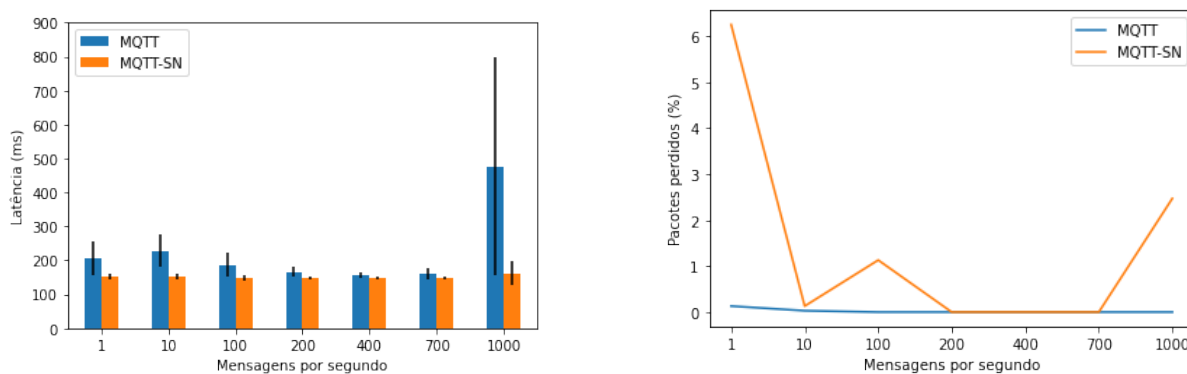


Figura 8 – Resultados para o cenário 4

Aqui encontra-se o primeiro caso fora da curva onde a latência média do MQTT chegou a 475 ms com 321 ms de desvio padrão, representando assim um possível gargalo.

4.1.5 Cenário 5 (Sub: 1, Pub: 16)

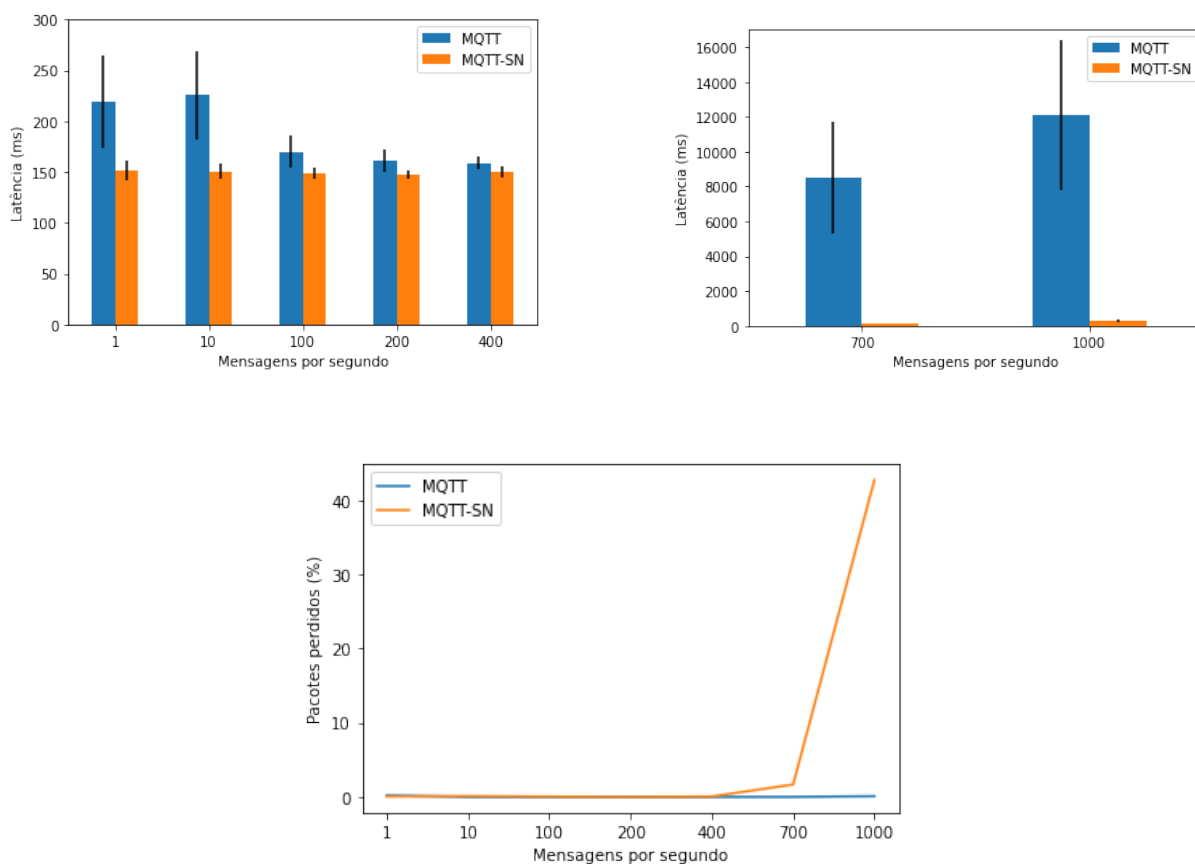


Figura 9 – Resultados para o cenário 5

Com 16 publicadores é visualizado graves atrasos na comunicação por parte do MQTT, com média de 8535 ms e 12115 ms para 700 msg/s e 1000 msg/s respectivamente. Para os mesmos parâmetros o desvio padrão foi de 3195 ms e 4274 ms respectivamente. MQTT-SN sofreu menores impactos na latência, um leve aumento para 700 msg/s com 155 ms e consideravelmente para 1000 msg/s com 302 ms.

Outra grande diferença está na quantidade de pacotes perdidos, MQTT teve máximo de 0,18% para 1 msg/s. MQTT-SN, por sua vez, teve perda de 1,67% para 700 msg/s e elevados 42,73% para 1000 msg/s.

4.1.6 Cenário 6 (Sub: 2, Pub: 1)

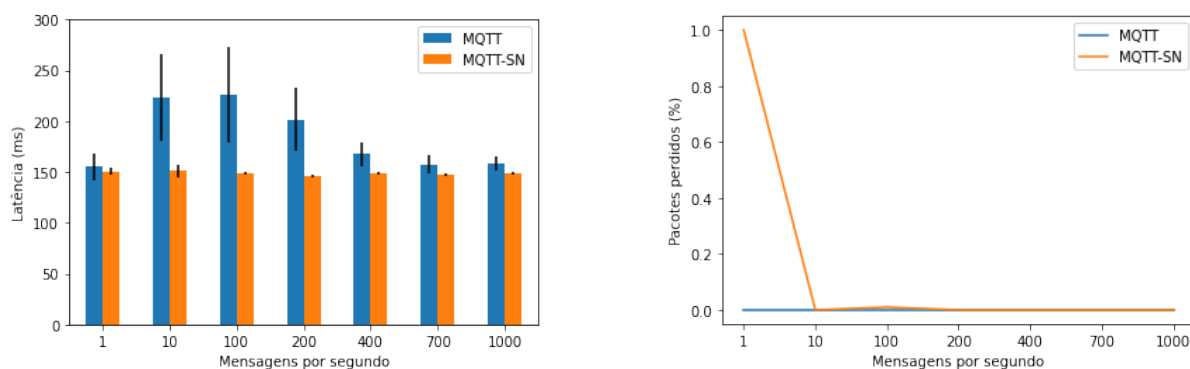


Figura 10 – Resultados para o cenário 6

Configuração semelhante ao cenário 2, apenas alterando de 2 publicadores para 2 assinantes. Os resultados também foram semelhantes.

4.1.7 Cenário 7 (Sub: 4, Pub: 1)

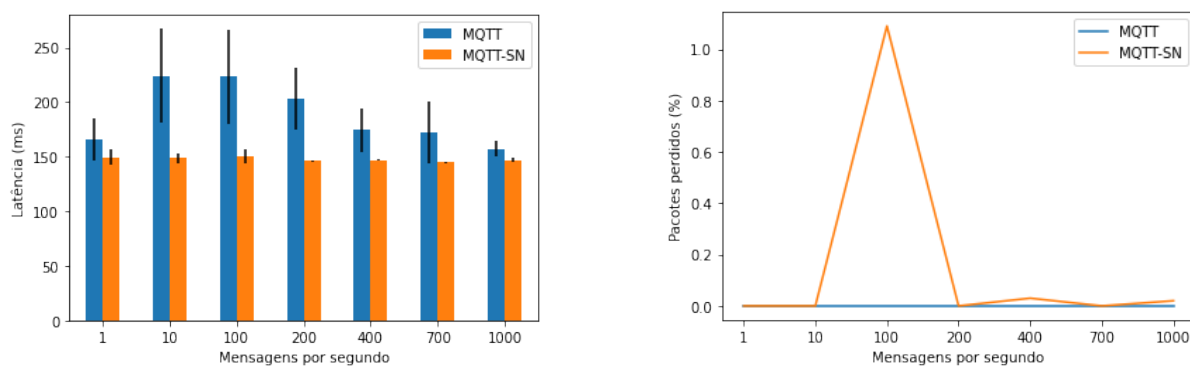


Figura 11 – Resultados para o cenário 7

Cenário sem grandes mudanças, métricas permaneceram estáveis em geral.

4.1.8 Cenário 8 (Sub: 8, Pub: 1)

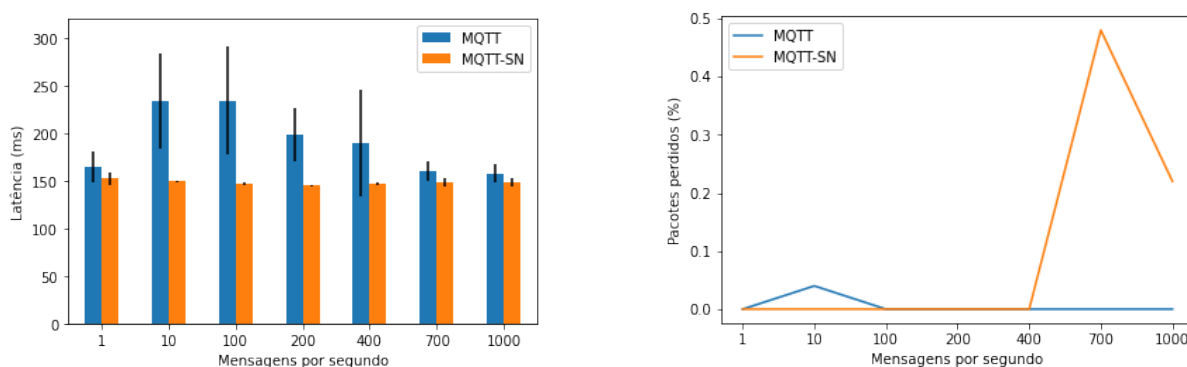


Figura 12 – Resultados para o cenário 8

Outro cenário com resultados estáveis, sem alterações. O desvio padrão da latência do MQTT atingiu máximo de 6.52 ms para 1 msg/s e mínimo 0.86ms para 10 msg/s. Pacotes perdidos por sua vez representaram máximo de 0,5% no pior caso.

4.1.9 Cenário 9 (Sub: 16, Pub: 1)

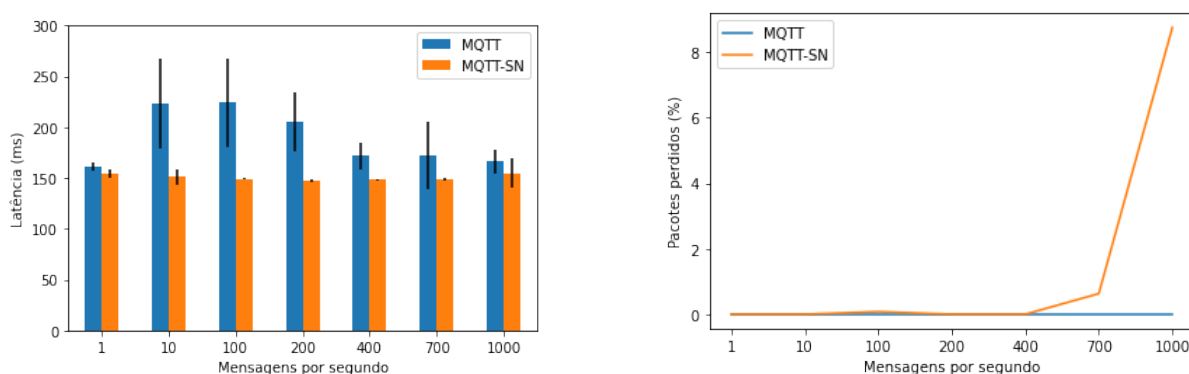


Figura 13 – Resultados para o cenário 9

Contraopondo-se aos resultados do cenário 5, apesar da mesma quantidade de transmissões, observou-se atrasos ligeiramente maiores do usual. As perdas de pacote foram inexistente para MQTT e de no máximo 8,75% para o MQTT-SN com 1000 msg/s.

4.1.10 Cenário 10 (Sub: 2, Pub: 2)

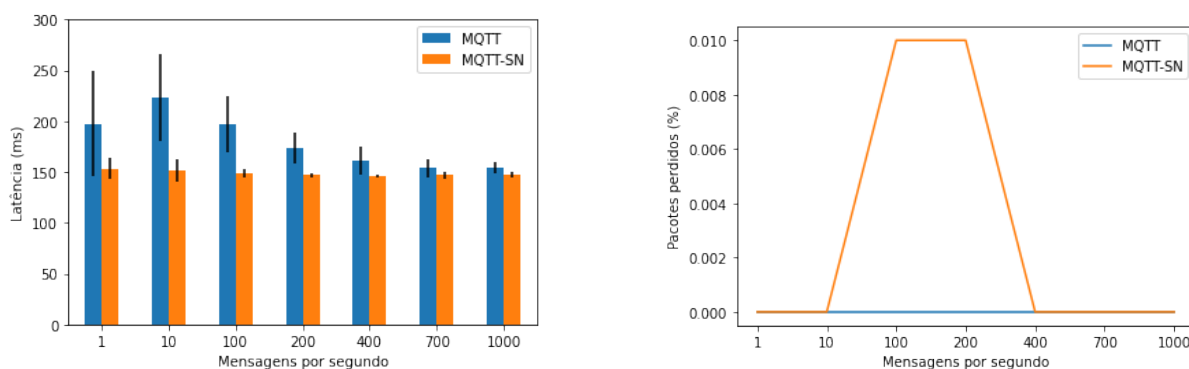


Figura 14 – Resultados para o cenário 10

Primeiro cenário usando cardinalidade N:N e, mesmo com mais carga, o fluxo de mensagens ainda não é intenso. Com intervalo de 4 msg/s até 4 mil msg/s não apresentou nenhuma forma de gargalo.

4.1.11 Cenário 11 (Sub: 4, Pub: 4)

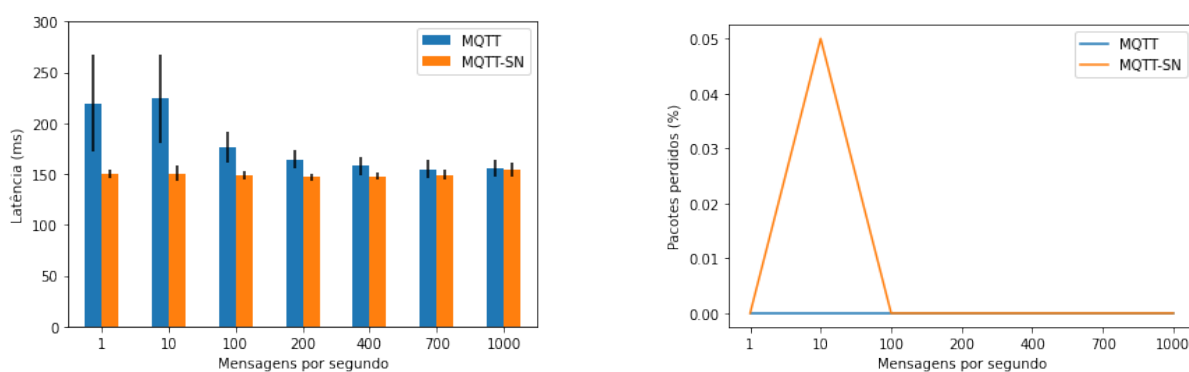


Figura 15 – Resultados para o cenário 11

Entre todos os cenários aqui encontramos a maior simetria na latência entre os dois protocolos, com exceção para 1, 10 e 100 msg/s. Os pacotes perdidos foram mínimos com apenas 0.05% no pior caso.

4.1.12 Cenário 12 (Sub: 8, Pub: 8)

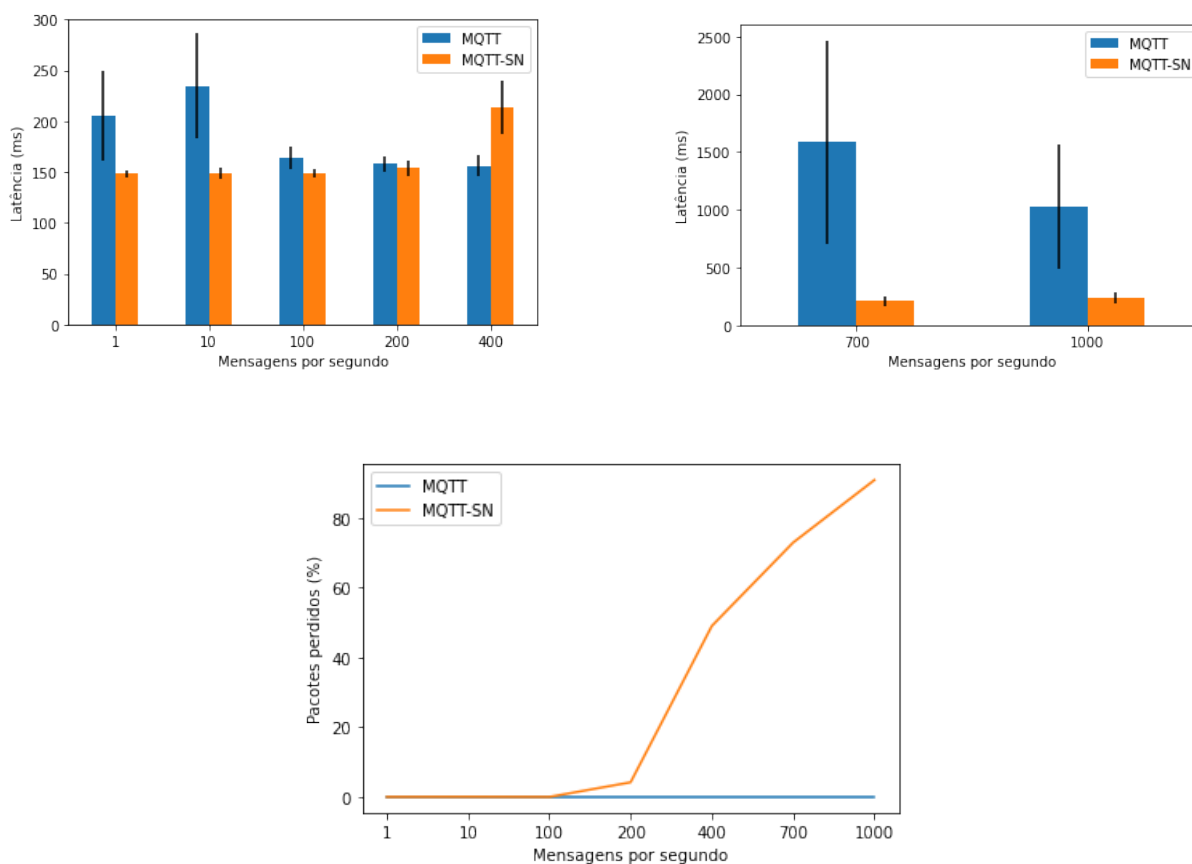


Figura 16 – Resultados para o cenário 12

Aqui encontramos um grande gargalo na transmissão de dados. Já com 200 msg/s, o MQTT-SN teve 4,19% de seus pacotes perdidos. Para 400, 700 e 1000 msg/s ocorreram 49,06%, 72,98% e 90,92% de perdas respectivamente.

Para a latência é notado um aumento de cerca de 50 ms para os parâmetros mais exigentes do MQTT-SN. O MQTT tem um atraso consideravelmente maior chegando a 1352 ms com 700 msg/s.

4.1.13 Cenário 13 (Sub: 16, Pub: 16)

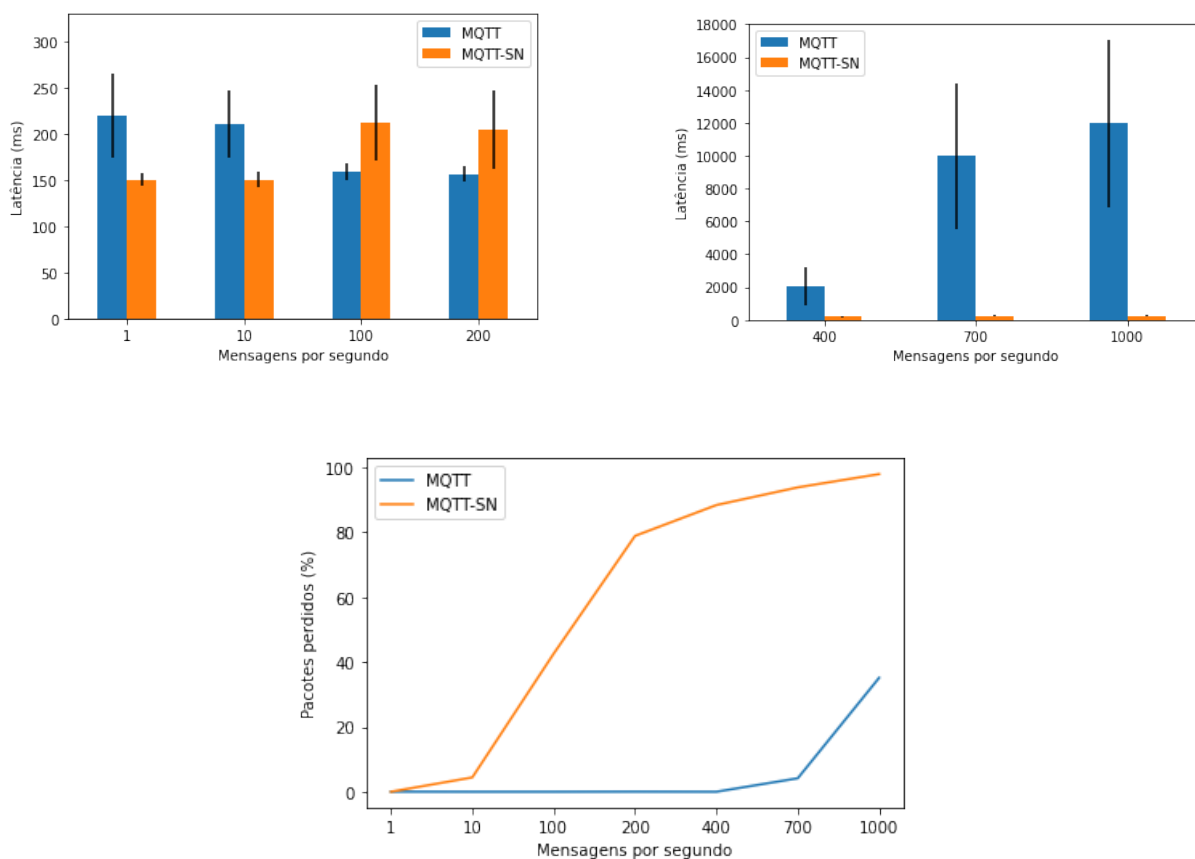


Figura 17 – Resultados para o cenário 13

Cenário com maior exigência de recursos, deixa claro a existência de gargalo. Isso pode ser pontuado tanto com a latência quanto com a perda de pacotes.

MQTT apresenta seu característico aumento na latência para evitar perdas de pacotes, apesar de não tão efetivo para 1000 msg/s onde 35,13% foram perdidos.

MQTT-SN por sua vez chegou a máximos 254,11 ms de latência média máxima para 700 msg/s. Sua limitação está na perda de pacotes que atingiu 97,9% no pior caso.

4.1.14 Cenário 14 (Sub: 16, Pub: 16, AWS)

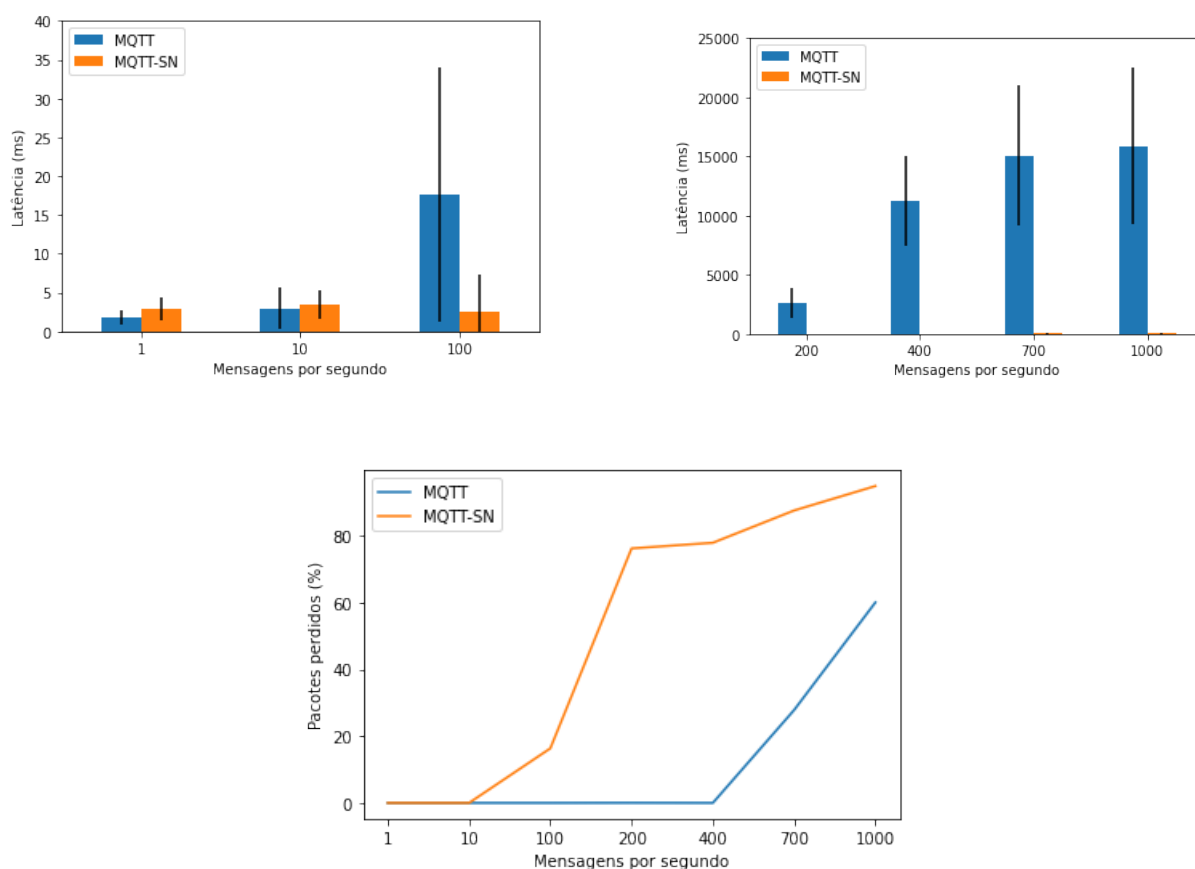


Figura 18 – Resultados para o cenário 14

Esse cenário difere dos demais, consiste em posicionar os clientes em uma máquina AWS EC2 com as mesmas características e região do broker. O objetivo é validar se o gargalo está ocorrendo na máquina local. Os parâmetros são iguais ao do cenário 13.

A expectativa de resultados semelhantes ao cenário 13, por usar os mesmos parâmetros se confirmou. Inclusive desempenharam levemente pior em todas as métricas.

4.2 ANÁLISE DOS RESULTADOS

É notável a diferença entre os protocolos, principalmente na forma como reagem nas dificuldades. MQTT por usar TCP irá buscar retransmitir mensagens que foram perdidas. Essas retransmissões demandam mais tempo aumentando assim a latência. O MQTT-SN usando UDP irá partir para a prática do melhor esforço, tentará entregar uma única vez mesmo caso aconteça a perda dessa mensagem.

Em situações onde há poucas mensagens sendo trafegadas (1, 10, 100), independente da quantidade de clientes, o tempo de latência foi próximo ao *Round Trip Time* e as perdas foram nulas ou próximas de zero.

Uma característica observada é a diferença ao inverter a cardinalidade entre publicadores e assinantes de 1:N para N:1. Ao aumentar a quantidade de publicadores, temos um atraso/perdas maiores do que ao aumentar a quantidade de assinantes. Isso é explicado pois com múltiplos assinantes as mensagens são replicadas apenas no nosso broker, diminuindo assim a chance de perdas no caminho.

As oscilações ficam frequentes quando temos cerca de 8 mil mensagens por segundo chegando a um mesmo tópico em nosso broker e, a partir disso, tende a ocorrer maiores atrasos e perdas.

O presente trabalho confirma o que foi apresentado por Chen; Kunz (6), onde o MQTT apresenta um aumento na latência conforme incrementa a necessidade de recursos. As evidentes perdas de pacotes do MQTT-SN reforça o estudo de Cosmi; Mota (7), onde apontaram a confiabilidade de entrega como um ponto negativo ao protocolo, porém não foi utilizado níveis de qualidade de serviço superiores a 0 que, a princípio, devem resolver esse problema.

Podemos corroborar com o trabalho de Mun; Le Dinh; Kwon (19), onde o MQTT-SN apresentou péssimos resultados de desempenho na transmissão de dados que foi justificado por uma implementação deficiente da biblioteca Paho mqttsn. Apesar de ter algumas limitações nas funcionalidades, o uso do projeto MQTT-SN Tools (15) provou-se uma boa alternativa.

A configuração de ambiente feita por Mun; Le Dinh; Kwon (19) mostrou-se eficaz. O uso de hospedagem em nuvem fornece a possibilidade de testarmos nossa aplicação em diversos pontos do globo terrestre verificando os efeitos disso.

5 CONCLUSÃO

Este trabalho apresentou a definição de cenários e análise do desempenho dos protocolos MQTT e MQTT-SN. Os critérios de avaliação são baseados nas métricas de latência e perda de pacotes. A proposta exaustão no tráfego de dados foi alcançada principalmente usando configuração máxima de clientes e de mensagens.

O paradigma publicador/assinante firma-se como uma opção para comunicação de dispositivos limitados de recursos. A necessidade de um broker centralizador é facilitada pela computação em nuvem que fornece máquinas de alto desempenho e com preço acessível.

Percebe-se que os protocolos cumprem o que é proposto. MQTT-SN pode ser usado em aplicações onde não há sensibilidade a perdas de mensagens. Como exemplo, podemos citar fazendas inteligentes, onde são coletadas amostras pluviométricas periodicamente. Em compensação, o MQTT resolve esse problema realizando retransmissões de pacotes proporcionando confiabilidade, apesar de consumir mais recursos de processamento e energia.

A lacuna presente nos trabalhos acadêmicos com foco em avaliação de protocolos para IoT estará mais próxima de ser preenchida. Os parâmetros utilizados provaram-se efetivos para definir limites de tráfego até que a aplicação apresente comportamentos indesejados.

No geral, o resultado do trabalho foi satisfatório, os objetivos propostos foram alcançados. Espera-se que isso ajude futuros profissionais do campo de Internet das Coisas na escolha do protocolo que melhor atende suas necessidades.

5.1 TRABALHOS FUTUROS

O presente trabalho foi restrito em avaliar algumas das principais métricas em ambientes limitados; no entanto, consumo de bateria, processamento e largura de rede não foram considerados. Recomenda-se aprofundar os estudos nessas métricas.

Outros parâmetros passíveis de serem alterados são o nível de qualidade de serviço e a largura de banda da rede. Aproveitando a mesma direção da atual pesquisa pode ser escalado o número de publicadores/assinantes com um baixo número de mensagens enviadas.

Uma continuação desejada é configurar uma aplicação real com objetos dispostos fisicamente dispersos, coletando informações fieis ao ambiente.

REFERÊNCIAS

- 1 AMARAN, Muhammad Harith et al. A comparison of lightweight communication protocols in robotic applications. **Procedia Computer Science**, Elsevier, v. 76, p. 400–405, 2015.
- 2 AURES, Georg; LÜBBEN, Christian. DDS vs. MQTT vs. VSL for IoT. **Network**, v. 1, 2019.
- 3 BANKS, Andrew et al. **MQTT v5.0 standard**. English. Versão Version 5.0. [S.l.], 2019. 137 p.
- 4 CHANG, Chii; SRIRAMA, Satish Narayana; BUYYA, Rajkumar. Internet of Things (IoT) and new computing paradigms. **Fog and edge computing: principles and paradigms**, Springer, v. 6, p. 1–23, 2019.
- 5 CHAUDHARY, Ajay; PEDDOJU, Sateesh K; KADARLA, Kavitha. Study of internet-of-things messaging protocols used for exchanging data with external sources. In: IEEE. 2017 IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS). [S.l.: s.n.], 2017. p. 666–671.
- 6 CHEN, Yuang; KUNZ, Thomas. Performance evaluation of IoT protocols under a constrained wireless access network. In: IEEE. 2016 International Conference on Selected Topics in Mobile & Wireless Networking (MoWNeT). [S.l.: s.n.], 2016. p. 1–7.
- 7 COSMI, Arthur Brito; MOTA, Vinicius FS. Uma Análise dos Protocolos de Comunicação para Internet das Coisas. In: SBC. ANAIS do III Workshop de Computação Urbana. [S.l.: s.n.], 2019. p. 153–166.
- 8 ECLIPSE, Foundation. **Eclipse Mosquitto**.
- 9 _____. **Eclipse Paho MQTT Go client**. 2016.
- 10 _____. **RSMB: Really Small Message Broker**. 2013.
- 11 FAQ - Frequently Asked Questions: MQTT. [S.l.: s.n.]. Disponível em: <<http://mqtt.org/faq>>.
- 12 AL-FUQAHA, Ala et al. Internet of things: A survey on enabling technologies, protocols, and applications. **IEEE communications surveys & tutorials**, IEEE, v. 17, n. 4, p. 2347–2376, 2015.
- 13 GARTNER 2020 Hype Cycle for Supply Chain Strategy Shows Internet of Things is Two to Five Years Away from Transformational Impact. [S.l.: s.n.], set. 2020. Disponível em: <<https://www.gartner.com/en/newsroom/press-releases/2020-09-09-gartner-2020-hype-cycle-for-supply-chain-strategy-shows-internet-of-things-is-two-to-five-years-away-from-transformational-impact>>.

- 14 GLOBAL Internet of Things Market (IoT Market) By Software Solution, By Platform, By Application, By Geographic Scope And Forecast. [S.l.: s.n.], jul. 2020. Disponível em: <<https://www.verifiedmarketresearch.com/product/global-internet-of-things-iot-market-size-and-forecast-to-2026/>>.
- 15 HUMFREY, Nicholas. **MQTT-SN Tools**. 2013.
- 16 HUNKELER, Urs; TRUONG, Hong Linh; STANFORD-CLARK, Andy. MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks. In: IEEE. 2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08). [S.l.: s.n.], 2008. p. 791–798.
- 17 INTERNET OF THINGS (IOT) MARKET - GROWTH, TRENDS, FORECASTS (2020 - 2025). [S.l.: s.n.]. Disponível em: <<https://www.mordorintelligence.com/industry-reports/internet-of-things-moving-towards-a-smarter-tomorrow-market-industry>>.
- 18 KARAGIANNIS, Vasileios et al. A survey on application layer protocols for the internet of things. **Transaction on IoT and Cloud computing**, v. 3, n. 1, p. 11–17, 2015.
- 19 MUN, Dae-Hyeok; LE DINH, Minh; KWON, Young-Woo. An assessment of internet of things protocols for resource-constrained applications. In: IEEE. 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC). [S.l.: s.n.], 2016. v. 1, p. 555–560.
- 20 NAIK, Nitin. Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In: IEEE. 2017 IEEE international systems engineering symposium (ISSE). [S.l.: s.n.], 2017. p. 1–7.
- 21 PARDO-CASTELLOTE, Gerardo. Omg data-distribution service: Architectural overview. In: IEEE. 23RD International Conference on Distributed Computing Systems Workshops, 2003. Proceedings. [S.l.: s.n.], 2003. p. 200–206.
- 22 QUINCOZES, Silvio; EMILIO, Tubino; KAZIENKO, Juliano. MQTT Protocol: Fundamentals, Tools and Future Directions. **IEEE Latin America Transactions**, IEEE, v. 17, n. 09, p. 1439–1448, 2019.
- 23 SANTOS, Bruno P et al. Internet das coisas: da teoria à prática. **Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos**, v. 31, 2016.
- 24 SCHÜTZ, Bertram; BAUER, Jan; ASCHENBRUCK, Nils. Improving energy efficiency of MQTT-SN in lossy environments using seed-based network coding. In: IEEE. 2017 IEEE 42nd Conference on Local Computer Networks (LCN). [S.l.: s.n.], 2017. p. 286–293.

- 25 SILVA, Edelberto Franco et al. IEEE P21451-1-7: Providing more efficient network services over MQTT-SN. In: IEEE. 2019 IEEE Sensors Applications Symposium (SAS). [S.l.: s.n.], 2019. p. 1–5.
- 26 SUBRAMONI, Hari et al. Design and evaluation of benchmarks for financial applications using Advanced Message Queuing Protocol (AMQP) over InfiniBand. In: IEEE. 2008 Workshop on High Performance Computational Finance. [S.l.: s.n.], 2008. p. 1–8.
- 27 TALAMINOS-BARROSO, Alejandro et al. A Machine-to-Machine protocol benchmark for eHealth applications–Use case: Respiratory rehabilitation. **Computer methods and programs in biomedicine**, Elsevier, v. 129, p. 1–11, 2016.
- 28 VIRIATO, Vando; MARTIN; CONCEICAO, Jose Da. **Introduction to MQTT-SN (MQTT for Sensor Networks)**. [S.l.: s.n.]. Disponível em:
<<http://www.steves-internet-guide.com/mqtt-sn/>>.
- 29 YASSEIN, Muneer Bani et al. Internet of Things: Survey and open issues of MQTT protocol. In: IEEE. 2017 International Conference on Engineering & MIS (ICEMIS). [S.l.: s.n.], 2017. p. 1–6.