UNIVERSIDADE DO ESTADO DO AMAZONAS
ESCOLA SUPERIOR DE TECNOLOGIA
CURSO DE ENGENHARIA ELÉTRICA

STHEFANY DO SOCORRO SOUZA DA SILVA

ANÁLISE DE DESEMPENHO DO PROTOCOLO MQTT-SN NO PADRÃO IEEE 802.11 UTILIZANDO O SIMULADOR OMNET++ PARA APLICAÇÕES IOT

MANAUS
2024

STHEFANY DO SOCORRO SOUZA DA SILVA

ANÁLISE DE DESEMPENHO DO PROTOCOLO MQTT-SN NO PADRÃO IEEE 802.11 UTILIZANDO O SIMULADOR OMNET++ PARA APLICAÇÕES IOT

Monografia apresentada ao curso de graduação em Engenharia Elétrica da Escola Superior de Tecnologia (EST) da Universidade do Estado do Amazonas (UEA), para obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Prof. Rubens de Andrade Fernandes, D.Sc.

MANAUS

2024

Universidade do Estado do Amazonas – UEA
Escola Superior de Tecnologia - EST

Reitor:
André Luiz Nunes Zogahib
Vice-Reitora:
Kátia Nascimento Couceiro
Diretor da Escola Superior de Tecnologia:
Jucimar Maia da Silva Junior
Coordenador do Curso de Engenharia Elétrica:
Jozias Parente de Oliveira

Banca Avaliadora composta por: *Data da defesa: 13/12/2024.*
Prof. Rubens de Andrade Fernandes, D.Sc. *(Orientador)*
Prof. Antonio Luiz Alencar Pantoja, Esp.
Prof. Jozias Parente de Oliveira, D.Sc.

CIP – Catalogação na Publicação

Da Silva, Sthefany do Socorro Souza

análise de desempenho do protocolo MQTT-SN no padrão IEEE 802.11 utilizando o simulador OMNeT++ para aplicações IoT / Sthefany do Socorro Souza da Silva; [orientado por] Rubens Fernandes de Andrade. – Manaus: 2024.

61p.: il.

Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica). Universidade do Estado do Amazonas, 2024.

1. Internet of Things. 2. Protocolos de Comunicação. 3. MQTT-SN. Fernandes, Rubens de Andrade.

STHEFANY DO SOCORRO SOUZA DA SILVA

ANÁLISE DE DESEMPENHO DO PROTOCOLO MQTT-SN NO PADRÃO IEEE 802.11 UTILIZANDO O SIMULADOR OMNET++ PARA APLICAÇÕES IOT

Pesquisa desenvolvida durante a disciplina de Trabalho de Conclusão de Curso II e apresentada à banca avaliadora do Curso de Engenharia Elétrica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas, como pré-requisito para a obtenção do título de bacharel.

Nota obtida: 10,0 (Dez vírgula zero)

Aprovada em 13 / 12 / 2024.

Área de concentração: Redes de Comunicações

BANCA EXAMINADORA

Orientador: Prof. Rubens Fernandes de Andrade, D.Sc. (UEA)

Avaliador: Antônio Luiz Alencar Pantoja, Esp. (UEA)

Avaliador: Jozias Parente de Oliveira, D.Sc. (UEA)

Manaus, 2024

AGRADECIMENTOS

Por muito tempo, me tornar engenheira parecia algo inimaginável, distante e até inconcebível. Hoje, ao concluir essa jornada, posso dizer com orgulho que realizei esse sonho. Foram cinco anos marcados por esforço, trabalho duro e resiliência, e sou profundamente grata à Sthefany de 2019, que teve a coragem de entrar no desconhecido e, com isso, permitiu que eu chegasse até aqui. No entanto, nenhum caminho é trilhado sozinho. Se hoje estou aqui, é porque tive pessoas que me apoiaram, me encorajaram e estiveram ao meu lado em cada passo.

Primeiramente, agradeço à minha família. Aos meus pais, Alessandra Souza e Francisco Silva, que me criaram para ser mais do que qualquer texto poderia descrever, que me apoiaram quando duvidei de mim mesma e sempre me ofereceram um lugar seguro para onde eu poderia voltar, independentemente das circunstâncias. Às minhas irmãs, Evelyn Silva e Lavinia Silva, que tornaram a vida mais leve e divertida, dividiram comigo noites em claro e ajudaram a elucidar minhas ideias quando eu me perdia pelo caminho. Agradeço também à Família Munck, que ao longo da vida me ofereceram apoio de tantas formas e que considero um presente divino.

Expresso minha profunda gratidão ao meu namorado, melhor amigo e parceiro, Kaio Lima, que esteve ao meu lado, literalmente, nos momentos em que parecia impossível avançar com esse trabalho. Que acreditou na minha capacidade e me mostrou que é possível encontrar conforto e força em meio ao caos da vida adulta. Seu apoio foi essencial e por ele, sou grata.

Agradeço também aos amigos que a faculdade me deu. Ao grupo *Los Cornitos* — Bruno Pantoja, Caroline Yokota, Mauricio Valente (*in memoriam*), Paulo Torres e Wagner Farah — que trouxeram leveza aos dias mais difíceis, transformaram frustrações em risadas e toparam enfrentar comigo todos os desafios que surgiram ao longo do curso. Às *Engmanas*, Laura Sobral, Gabriela Damasceno, que entraram de cabeça comigo nesse projeto que inspira tantas meninas, e que ressignificaram a força e apoio da amizade feminina no mundo da Engenharia, e que continuam a ser uma inspiração, destacando-se em suas carreiras e em suas vidas. Em especial, agradeço à primeira a acreditar nesse projeto, minha grande amiga e confidente, Maysa Marques, que esteve ao meu lado em momentos de insegurança e sempre me deu coragem para enfrentar novos desafios, sendo uma inspiração para sempre lembrar de fazer o que quisermos.

Finalmente, com imensa gratidão, agradeço ao meu orientador, Rubens Fernandes, que acreditou neste tema e me orientou com tanta paciência e dedicação.

Menciono também todos aqueles que, direta ou indiretamente, estiveram presentes nesta jornada alguma forma. Cada gesto, palavra de encorajamento e presença foi fundamental para que eu alcançasse este momento. A todos vocês, minha mais sincera gratidão.

Deseje e sonhe com muita fé no coração, mas lembre-se que aquela estrelinha só é responsável pela metade. O resto você faz com muito trabalho, e então aí sim, você vai poder fazer tudo o que imaginar.

A princesa e o Sapo

RESUMO

A *Internet of Things* (IoT) refere-se a uma rede composta por elementos como comunicação, sensores e serviços, que possibilitam a navegação na internet, a troca de informações e a comunicação entre dispositivos. A eficiência dessa troca de informações depende de protocolos de comunicação, que impactam diretamente o desempenho dos dispositivos. Com a crescente demanda por dispositivos IoT em diversas aplicações, o desempenho da comunicação tornou-se um fator crítico devido ao aumento constante no volume de dados transmitidos.

Este trabalho teve como objetivo analisar o desempenho do protocolo de comunicação MQTT-SN aplicado a dispositivos IoT. Para isso, foram estabelecidos métricas de interesse e realizadas simulações de cenários de aplicação de sistemas IoT, como alta frequência de publicação, aumento de publicadores e *broadcast*, no simulador de redes OMNeT++. Os resultados indicaram que a latência aumenta com níveis mais elevados de *Quality of Service* (QoS) e é impactada por sobrecargas na rede. No cenário com alta frequência de publicação, observou-se maior taxa de *overhead*, enquanto o cenário de *broadcast* apresentou menor latência e uma taxa de entrega de mensagens próxima a 100%. No cenário com aumento progressivo de publicadores, o *throughput* cresceu significativamente à medida que o número de publicadores aumentou. Esses resultados evidenciam a importância de equilibrar as exigências de QoS, os intervalos de publicação e inscrição, e a quantidade de dispositivos na rede para maximizar o desempenho da comunicação.

Palavras-chave: Internet of Things, protocolos de comunicação, MQTT-SN, simulação.

ABSTRACT

The Internet of Things (IoT) refers to a network composed of elements such as communication, sensors, and services that enable internet browsing, information exchange, and communication between devices. The efficiency of this information exchange depends on communication protocols, which directly impact the performance of devices. With the growing demand for IoT devices in various applications, communication performance has become a critical factor due to the constant increase in the volume of transmitted data.

This work aimed to analyze the performance of the MQTT-SN communication protocol applied to IoT devices. To this end, metrics of interest were established and simulations of IoT system application scenarios, such as high publishing frequency, increased publishers, and *broadcast*, were performed in the OMNeT++ network simulator. The results indicated that latency increases with higher levels of Quality of Service (QoS) and is impacted by network overloads. In the scenario with high publishing frequency, a higher overhead rate was observed, while the *broadcast* scenario presented lower latency and a message delivery rate close to 100%. In the scenario with progressive increase in publishers, throughput increased significantly as the number of publishers increased. These results highlight the importance of balancing QoS requirements, publication and subscription intervals, and the number of devices in the network to maximize communication performance.

Keywords: Internet of Things, communication protocols, MQTT-SN, simulation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

FTP	*File Transfer Protocol*
HTTP	*Hypertext Transfer Protocol*
IEEE	*Institute of Electrical and Electronic Engineers*
IoT	*Internet of Things*
IP	*Internet Protocol*
JSON	*JavaScript Object Notation*
MQTT	*Message Queuing Telemetry Transport*
MQTT-SN	*Message Queuing Telemetry Transport for Sensors Networks*
OMNeT++	*Objective Modular Network Testbed in C++*
QoS	*Quality of Service*
SMTP	*Simple Mail Transfer Protocol*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*

SUMÁRIO

1 INTRODUÇÃO

Com a evolução constante da tecnologia, a relação homem-máquina se desenvolveu de tal forma que as máquinas fazem parte do cotidiano do ser humano, de forma intrínseca. Isso se manifesta não apenas em dispositivos tradicionais, como celulares e computadores, mas também na crescente presença de dispositivos inteligentes interconectados por meio da internet, utilizando conceitos de *Internet of Things* (IoT) (Leite, Martins e Ursini, 2017).

A Internet das Coisas, mais conhecida como *Internet of Things*, se refere a paradigma que permite a conexão, comunicação e troca de dados entre objetos físicos ou dispositivos, coletando e compartilhando informações por meio da internet (Minerva, Biru e Rotondi, 2015). Esses dispositivos inteligentes possuem a capacidade de comunicação e processamento aliados a sensores, atuadores e tecnologia de comunicação, e fornecem informações do ambiente ao seu redor ao interagir com outros dispositivos para realizar ações específicas, criando um ambiente inteligente e eficiente, onde estes dispositivos possam ser monitorados, controlados e otimizados remotamente (Godoi e Araújo, 2019). A *Internet of Things* é um tema de crescente importância no desenvolvimento tecnológico contemporâneo e está presente em diversas áreas, desde a agricultura até automação industrial.

Dessa maneira, a comunicação é um dos blocos essenciais de um sistema de Internet das Coisas e pode ser viabilizada através de conexões sem fio (*wireless)* ou com fio. A comunicação *wireless* permite que a transmissão de dados seja feita sem a necessidade cabos de conexão, como o Wi-Fi, por exemplo, que é uma tecnologia comumente difundida entre dispositivos eletrônicos com acesso à Internet através de ondas de rádio. Para viabilizar essa comunicação sem fio, são utilizados protocolos de comunicação. Os protocolos de comunicação para IoT, como *Message Queuing Telemetry Transport (*MQTT*)* são projetados para garantir a transmissão eficiente de dados entre dispositivos IoT e a nuvem, onde as informações ficam armazenadas. Eles funcionam como um conjunto de regras e padrões e definem os métodos possíveis para viabilizar a transmissão de dados (Cosmi e Mota, 2019).

No entanto, a comunicação eficiente entre dispositivos IoT continua sendo um desafio, especialmente em cenários que exigem transmissão contínua de dados, como redes Wi-Fi e ambientes remotos. A eficiência do protocolo de comunicação utilizado torna-se crucial para garantir uma comunicação robusta e eficiente, sobretudo em situações de conectividade limitada e alta demanda de transmissão de informações. Considerando que os meios de comunicação e transmissão de dados são os principais pilares de um sistema IoT, torna-se

essencial a avaliação do desempenho do protocolo de comunicação em diversos cenários, de acordo o objetivo geral do projeto.

Dentro deste contexto, observa-se a importância de análise das tecnologias de comunicação em um sistema de *Internet of Things*, como os protocolos de comunicação, assim, a proposta dessa pesquisa é realizar uma análise do desempenho do protocolo de comunicação de um sistema de IoT, a partir de simulações, quando aplicados diferentes cenários de simulação, permitindo assim, analisar o comportamento de protocolos em relação à diversas métricas de qualidade. O protocolo a ser analisado será o *MQTT for Sensors Networks* (MQTT-SN) e para as simulações será utilizado o simulador de rede denominado *Objective Modular Network Testbed in C++* (OMNeT++), que é um ambiente de simulação de protocolos de comunicação, em C++, de código aberto, modular, utilizado principalmente para simulação de redes de comunicação. Dentro da simulação, a comunicação será feita via padrão IEEE 802.11.

Essa pesquisa tem como objetivo analisar o comportamento da rede com a utilização do protocolo MQTT-SN em diferentes cenários de aplicação e identificar limitações e vantagens do protocolo. Especificamente, objetiva-se analisar o estado da arte sobre o tema, reproduzir os cenários de testes em ambiente simulado e simular diferentes cenários para então, analisar os resultados de acordo com os critérios de qualidade estabelecidos na metodologia.

A organização do trabalho apresenta os conceitos teóricos na seção 2, onde o tópico 2.1 discorre sobre os conceitos relacionados ao IoT e em seguida, o tópico 2.2 apresenta os conceitos relacionados a redes de comunicação, como protocolos, padrão IEEE 802.11 e *Quality of Service* (QoS). Na seção 3 será descrita a metologia aplicada na pesquisa e finalmente, na seção 4, serão apresentados os resultados obtidos nas etapas experimentais.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta os principais conceitos que serão relacionados a este trabalho, sendo destacada cada contribuição para o contexto geral da monografia.

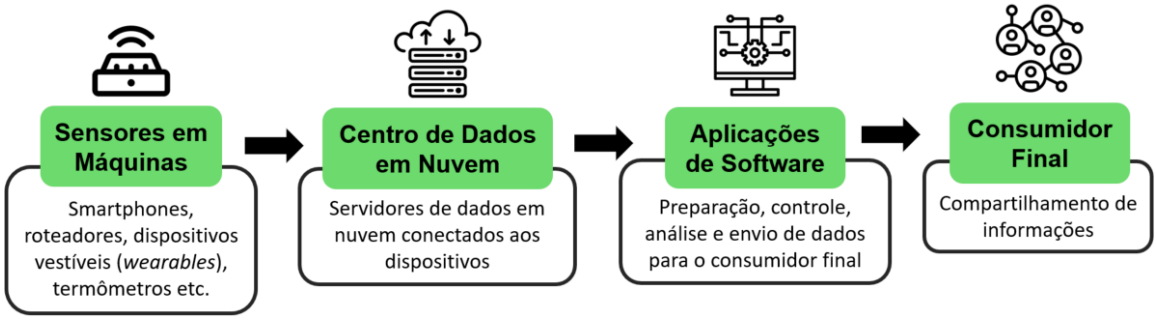
Na seção 2.1 será mostrado o conceito de Internet das Coisas. Em seguida, na seção 2.2, são expostos os conceitos base para redes de comunicação, onde se descrevem os conceitos relacionados a protocolos de comunicação, com ênfase aos protocolos para internet das coisas e qualidade de serviço.

2.1 Internet of Things (IoT)

Para Bassi, Europe e Horn (2008) o conceito de IoT está ligado à evolução tecnológica. Nesse sentido, computadores, sensores, telefones e dispositivos eletrônicos podem possuir endereços únicos que permitirão que eles possam navegar na internet, comunicarem entre si com identidades únicas e realizarem troca de informações com o intuito de facilitar o contexto social. Em suma, a IoT pode ser definida como um ambiente onde objetos físicos são interconectados na internet através de sensores que criam um ecossistema de comunicação, que tem como principal objetivo facilitar operações no dia a dia do ser humano e solucionar processos cotidianos (Magrani, 2018).

O processo de informações de um dispositivo IoT possui, cada um com uma finalidade específica (Figura 1).

Figura 1 - Fluxo de informação para dispositivos IoT.



Fonte: Moon, 2016 (Adaptado).

Os sensores que estão presentes nas máquinas são responsáveis por coletar dados disponíveis no ambiente ao seu redor. Esses dados são enviados para a nuvem, através da internet, para que possam ser analisados e repassados. Todo o processamento de dados é feito através de softwares para que possam chegar ao consumidor final (Moon, 2016).

As tendências tecnológicas relacionadas ao conceito de internet das coisas abrangem monitoramento e controle de dados, análise de dados massiva e compartilhamento de informações. Outras soluções mais recentes incluem cidades, carros e casas inteligentes, gerenciamento de energia e principalmente entretenimento (Medeiros, Colpo, *et al*., 2018).

2.2 Redes de comunicação

As redes de comunicação são sistemas que permitem a troca de informações entre dispositivos conectados, conhecidos como nós. Esses nós podem ser computadores, impressoras, roteadores, entre outros, e estão conectados por meio de links, que podem ser físicos ou sem fio. Durante a troca de informações, tem-se o *uplink* e o *downlink*. O *uplink* refere-se à conexão que envia dados de um dispositivo para a rede, enquanto o *downlink* é a conexão que recebe dados da rede para o dispositivo (Forouzan, 2010).

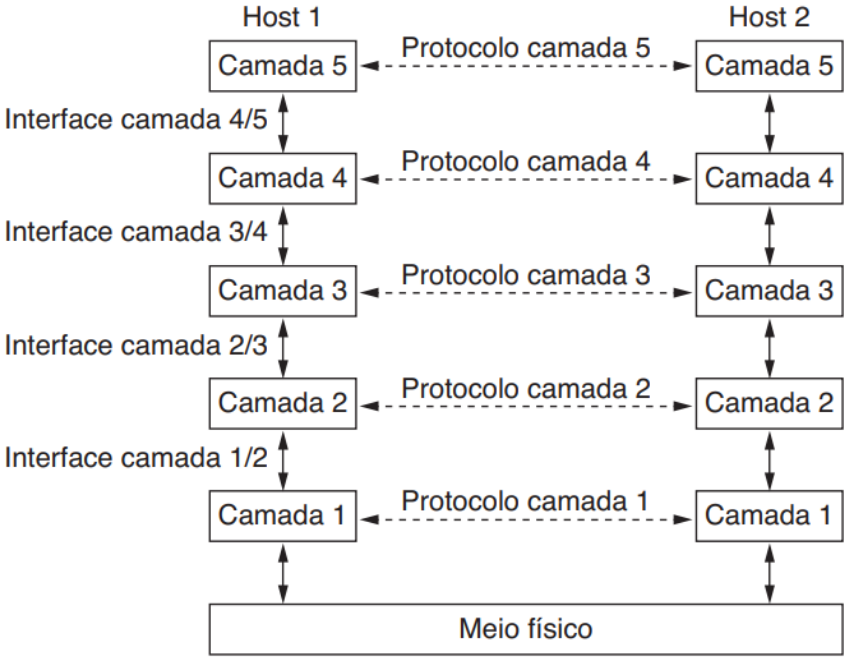
A conexão e a troca de informações em uma rede é governada por um conjunto de regras conhecido como protocolo de comunicação. Alguns exemplos de protocolos de comunicação incluem o *Hypertext Transfer Protocol* (HTTP), *Transmission Control Protocol/Internet Protocol* (TCP/IP) e *Simple Mail Transfer Protocol* (SMTP). A capacidade de uma rede transmitir dados é medida pela sua largura de banda, geralmente em bits por segundo (bps), enquanto a latência é o tempo que leva para os dados serem transferidos de um ponto a outro em uma rede (Tanenbaum e Wheterall, 2011).

2.2.1 Protocolos de comunicação

De acordo com Kurose e Ross (2013) protocolos de comunicação são conjuntos de regras que determinam como os dados são transmitidos e recebidos em uma rede. Eles definem aspectos como formato, sincronização, sequenciamento e verificação de erros, permitindo que dispositivos diferentes em uma rede se comuniquem de maneira eficaz.

Os protocolos podem operar em diferentes camadas de uma rede (Figura 2). Por exemplo, os protocolos de camada de aplicação, como HTTP e *File Transfer Protocol* (FTP), definem como os aplicativos em dispositivos de rede se comunicam entre si. Os protocolos de camada de transporte, como TCP e *User Datagram Protocol* (UDP), lidam com a transmissão confiável de dados entre dispositivos (Tanenbaum e Wheterall, 2011).

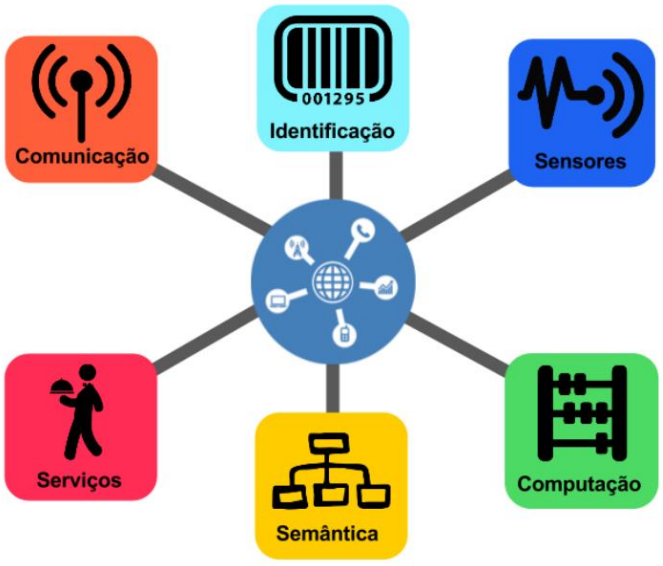
Figura 2 - Camadas, protocolos e interfaces de comunicação.



Fonte: Tanenbaum e Wheterall, 2011.

Conforme descrito na seção 2.1, a Internet das Coisas é um conjunto de tecnologias que se complementam para um objetivo final e pode-se dizer que a IoT é construída em blocos, conforme a Figura 3.

Figura 3 - Blocos básicos de IoT.



Fonte: Santos, Silva, *et al*., 2016.

De forma geral, o bloco de comunicação se refere a todas as técnicas utilizadas para conectar todos os dispositivos dentro da IoT, como os protocolos de comunicação. Esse bloco é responsável por facilitar todo o processo de envio de dados (Santos, Silva, *et al*., 2016).

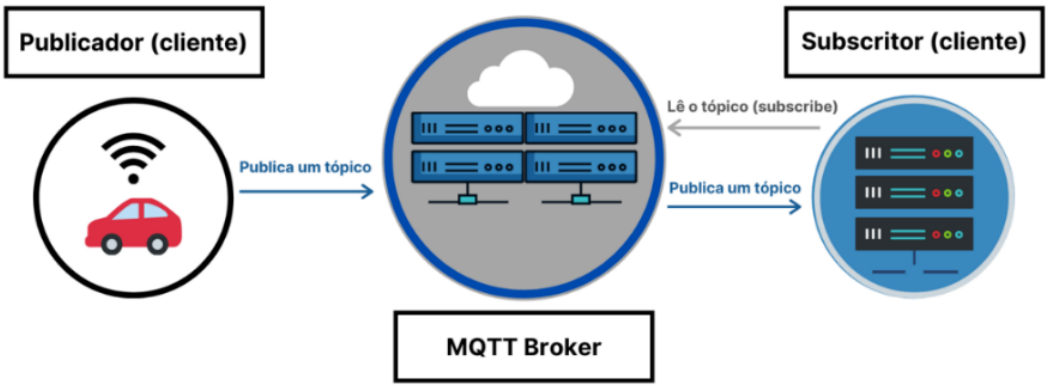
Os protocolos de comunicação são essenciais para os dispositivos IoT, uma vez que possibilitam que a parte de sensores se comunique e faça o envio de dados da melhor maneira possível, facilitando todo o processo (Macedo, Franciscatto, *et al*., 2018).

*2.2.1.1 Message Queuing Telemetry Transport (*MQTT*)*

Em 1999, Andy Stanford-Clark e Arlen Nipper desenvolveram o protocolo MQTT para conectar sistemas de telemetria via satélite que, na época, era um recurso de alto custo e por isso era necessário economizar banda e energia (Mishra e Kertesz, 2020).

O protocolo MQTT é um protocolo utilizado para envio de mensagens amplamente difundido em aplicações para IoT. O MQTT opera através de uma arquitetura do tipo cliente-servidor, isto é, os servidores como computadores de alto desempenho que armazenam e gerenciam dados enquanto os dispositivos (clientes) que solicitam serviços ou recursos do servidor. Nesse caso, o servidor é chamado de *broker*, que é o intermediário responsável por receber, encaminhar e distribuir as mensagens entre os clientes (Figura 4). Esses clientes podem publicar mensagens e outros clientes podem se inscrever em tópicos de seu interesse, quando isso ocorre, todos os clientes inscritos em um tópico irão receber uma mensagem permitindo que esses dispositivos se comuniquem de forma síncrona, mesmo que não estejam conectados diretamente (Yassein, Shatnawi, *et al.,* 2017).

Figura 4 - Exemplo de arquitetura para o protocolo MQTT.



Fonte: Beloti, 2020.

Durante a comunicação, esse protocolo é capaz de armazenar mensagens para que, quando o cliente se conecte, ele possa receber a última mensagem disponível, sendo esta uma das principais características do MQTT. Além disso, o MQTT permite que o cliente possa enviar mensagens específicas mesmo quando perde conexão (Soni e Makwana, 2017).

O MQTT é amplamente utilizado para aplicações em *Internet of Things* e permite a comunicação entre dispositivos, sensores e diversos itens essenciais na criação de um sistema desse tipo, oferecendo flexibilidade e escalabilidade, que facilita a sua aplicação em diversos cenários.

*2.2.1.2 MQTT for Sensors Networks (*MQTT-SN*)*

As inovações relacionadas aos sistemas de IoT possuem diversas fases de evolução e assim, os desafios se tornaram mais complexos, como preocupações com segurança e privacidade, armazenamento e nuvem, energia e comunicação. Nesse sentido, é essencial considerar como a comunicação é feita, uma vez que a transmissão de dados é extremamente variável, no sentido de pacotes de dados (pequenos e grandes), e em distância de transmissão (pequenas, médias e longas) (Pereira, Correia, *et al.*, 2020).

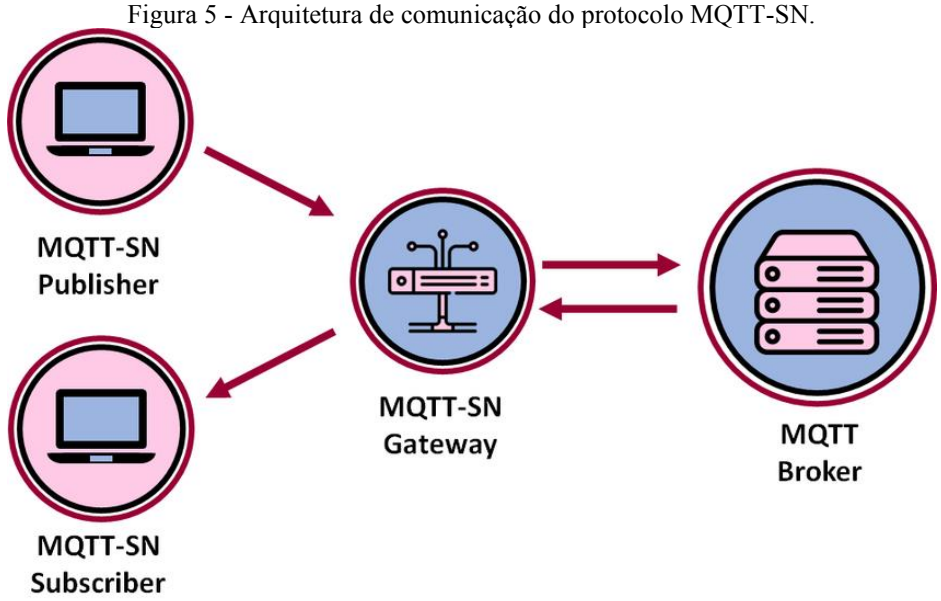
Segundo Zahoor e Mir (2018, p. 923), os recursos limitados são um aspecto importante a ser considerado na construção de sistemas IoT, como bateria, capacidade de processamento, largura de banda e consumo de energia. Ao se deparar com recursos limitados, os protocolos de comunicação estão evoluindo para realizar o processamento e envio de mensagens de forma mais precisa com o menor consumo de energia, além de ter entre seus objetivos principais uma maior eficiência de processamento.

Dessa maneira, o MQTT-SN foi desenvolvido com o objetivo para viabilizar a comunicação de dispositivos IoT em redes de sensores sem fio (*wireless*), que têm como característica o baixo poder computacional e a intenção de se manterem inativos por determinado tempo, a fim de economizar energia e serem alocados em locais remotos e/ou de difícil acesso (Quincozes, Tubino e Kazienko, 2019).

A principal diferença entre o MQTT e o MQTT-SN é a infraestrutura de comunicação, já que o MQTT-SN foi idealizado para trabalhar com comunicação por UDP. Assim, por se tratar de um protocolo não orientado a conexão, os dispositivos podem gerenciar suas interfaces de rede de forma dinâmica, de acordo com os seus objetivos. Além disso, o tamanho da mensagem foi reduzido para dar suporte a protocolos de camadas inferiores que possuem largura de banda limitada, como os que utilizam o padrão IEEE 802.15.4 (baixa taxa de transmissão). O MQTT-SN define seus tópicos com endereçamento por identificadores numéricos, eliminando a necessidade de manter uma conexão constante, diferentemente do MQTT padrão, que mantém o tópico em formato completo de texto, assim, essa configuração auxilia na redução do *overhead* de rede (Herrero, 2020).

O protocolo possui uma arquitetura de rede semelhante ao MQTT, com publicadores (*publishers*) e assinantes (*subscribers*), porém, na arquitetura é empregado um *gateway* (Figura 5) responsável pela interconexão dos dispositivos à rede e assim, a rede pode se comunicar de duas maneiras: transparente e agregada. Na comunicação transparente, cada cliente realiza em conexão com o *broker*, já de modo agregado, os clientes compartilham uma única conexão (Spohn e Genero, 2023).

Figura 5 - Arquitetura de comunicação do protocolo MQTT-SN.



Fonte: Palmase, Redondi e Cesana, 2022.

Sendo assim, o MQTT-SN se define como um protocolo de comunicação otimizado para dispositivos restritos, em armazenamento e consumo de energia, amplamente utilizado em aplicações IoT com sensores de baixo custo.

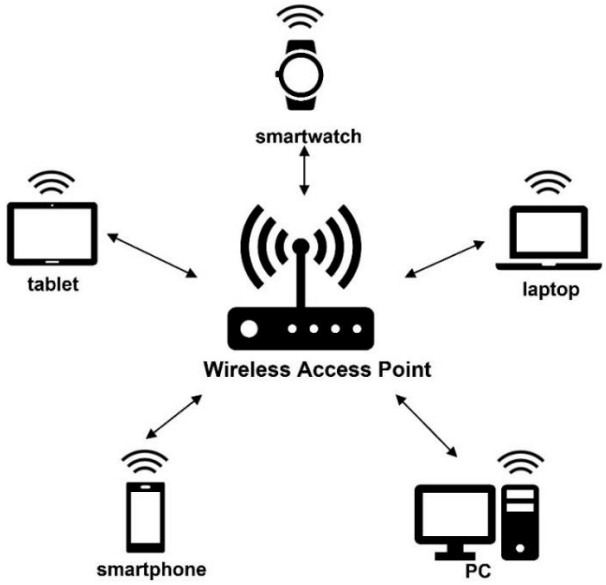
*2.2.2 Padrão IEEE 802.11 (*Wi-Fi*)*

O padrão IEEE 802.11 é o conjunto de normas e especificações desenvolvido pelo *Institute of Electrical and Electronic Engineers* (IEEE) para serem utilizadas em aplicações de redes locais sem fio, também nomeadas de Wi-Fi. Essas normas permitem a comunicação de dispositivos através de sinais de rádio, em contrapartida as redes cabeadas tradicionais. O Wi-Fi é amplamente utilizado em diferentes contextos, como redes domésticas, coorporativas, espaços públicos e, atualmente, em soluções de IoT (Lopez-Aguilera, Garcia-Villegas e Casademont, 2019).

Para aplicações de IoT, o padrão IEEE 802.11 demonstra-se vantajoso devido às suas características de largura de banda e o suporte a alta densidade de dispositivos. Em relação a

largura de banda, é possível transmitir grandes volumes de dados, como em casos de monitoramento de vídeo, principalmente pela possibilidade de operar nas frequências de 2,4 GHz e 5 GHz, possibilitando a escolha do canal de comunicação, por exemplo: a faixa de 2,4 GHz é indicada para aplicações em longa distância devido a maior penetração do sinal enquanto a faixa de 5 GHz possui uma maior velocidade de transmissão, indicada para aplicações com taxa alta de dados. Ainda, o padrão suporta diferentes modos de operação, como infraestrutura e ad hoc, sendo o modo de infraestrutura difundido em aplicações IoT, onde dispositivos se conectam por meio de um ponto de acesso (*Access Point*), conforme a Figura 6 (Khorov, Levitsky e Akyildiz, 2020).

Figura 6 - Possibilidades de conexão com o ponto de acesso.



Fonte: Yongatek Microeletronics.

No contexto desse trabalho, o padrão IEEE 802.11 é utilizado para viabilizar a comunicação sem fio entre os dispositivos que se comunicam através do protocolo MQTT-SN. Devido a sua natureza de aplicação para redes com recursos limitados, esse padrão fornece uma infraestrutura onde é viável analisar a escalabilidade e eficiência da comunicação.

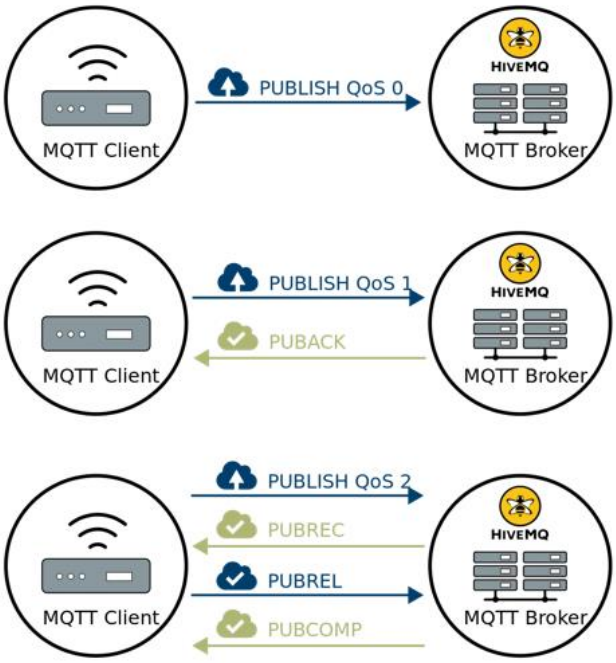
2.2.3 Quality of Service (QoS)

Singh e Baranwal (2018, p. 2) definem *Quality of Service* (QoS) ou Qualidade de Serviço, como o conjunto de tecnologias que gerenciam as capacidades de recursos de serviços de rede em relação a determinados requisitos de desempenho, como perdas de pacotes, largura de banda, latência, confiabilidade e eficiência de entrega. As métricas relacionadas a QoS auxiliam os consumidores e desenvolvedores a definirem qual a melhor abordagem para a sua

aplicação e como resolver problemas de otimização na qualidade de serviço. Em sistemas IoT, o QoS se torna um requisito de análise crucial devido às características intrínsecas ao sistema, como sensores, atuadores, limitações de recursos e alto tráfego de dados, além do ambiente onde as aplicações serão inseridas, como ambientes ruidosos, redes congestionadas e pouca capacidade de fornecimento de energia. Para realizar essa análise, o QoS define níveis que são usados para balancear o sistema e adaptar às características para melhores índices, sendo eles: Qos 0, QoS 1 e QoS 2 (Figura 7).

O nível de QoS 0 se refere ao "*At most once*" (uma vez). Nesse nível, a mensagem é enviada apenas uma vez e não é armazenada, além disso, não existe nenhuma confirmação para o seu recebimento. Assim, o consumo de energia e largura de banda são reduzidos, porém, é possível haver perdas nas mensagens. No nível de Qos 1, a mensagem é reenviada até que se receba alguma confirmação e nomeia-se como "*At least once"* (no mínimo uma vez). Nesse caso, a mensagem é armazenada até que a confirmação seja recebida e há a possibilidade de duplicação de informações. Por fim, no nível de QoS 2, tem-se o "*Exactly once*" (exatamente uma vez), onde existe a garantia de que a mensagem será entregue exatamente uma vez, evitando tanto a perda quanto a duplicação de mensagens (Toldinas, Lozinskis, *et al*., 2019).

Figura 7 - Comportamento de rede em diferentes níveis de QoS.



Fonte: HiveMQ, 2024.

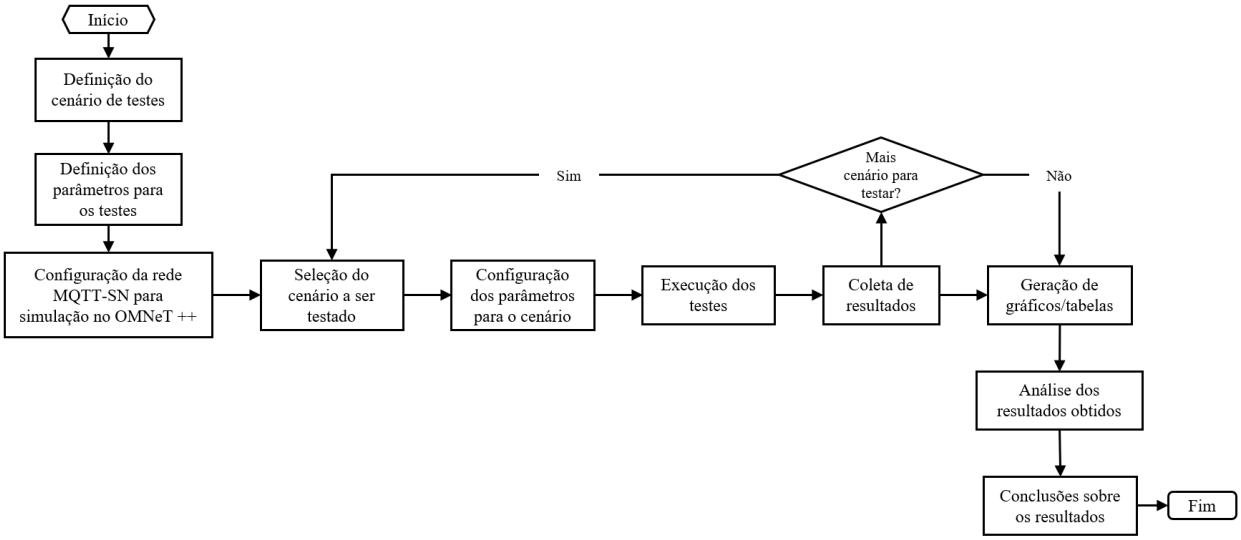
Tanto o MQTT quanto o MQTT-SN possuem a capacidade de variação de QoS na sua configuração. Dessa maneira, a escolha do nível de QoS deve ser feita com base nas

necessidades de aplicação, considerando os cenários disponíveis. Essas análises são essenciais para equilibrar eficiência, confiabilidade e manejar o uso de recursos.

3 MATERIAIS E MÉTODOS

De acordo com Severino (2013), o método de pesquisa deste trabalho foi definido como estudo de caso, sendo assim, o principal objetivo é revisar com profundidade aspectos de um determinado objeto de pesquisa, nesse caso, protocolos de comunicação. Além disso, a metodologia a ser aplicada se refere, em primeiro lugar, pela abordagem hipotético-dedutiva, sendo definida anteriormente a hipótese base do trabalho e, pelo procedimento, se refere ao procedimento monográfico, onde é realizado um estudo de um tema particular, analisando seus ângulos e aspectos. Para realizar os itens descritos, a pesquisa foi desenvolvida conforme a Figura 8.

Figura 8 - Fluxograma de atividades da metodologia.



Fonte: Autoria Própria.

Em primeiro lugar, definiu-se os cenários de testes para analisar o desempenho do protocolo MQTT-SN. Em seguida, foi ajustado o sistema de testes no simulador OMNeT++ conforme descrito nos tópicos a seguir. Assim, foram realizados os ajustes de parâmetros para a realização dos testes, para analisar pontos estabelecidos como critérios de estudo relevantes para essa pesquisa. A partir dos resultados dos testes realizados, foram feitas inferências para determinar as conclusões do trabalho de acordo com as definições das próximas seções. Por fim, as conclusões da pesquisa foram definidas com base nos resultados obtidos.

3.1 Ambiente de desenvolvimento

Para a realização dessa pesquisa será utilizado o hardware com a seguinte configuração (Figura 9): Processador Intel(R) Core(TM) i7-1165G7, 2,80 GHz; Memória RAM 8,00 GB; Sistema operacional de 64 bits, Windows 11.

Figura 9 - Notebook Samsung NP550XDA-KP3BR.

Fonte: Samsung Eletrônica da Amazônia LTDA.

3.2 Ferramentas de desenvolvimento

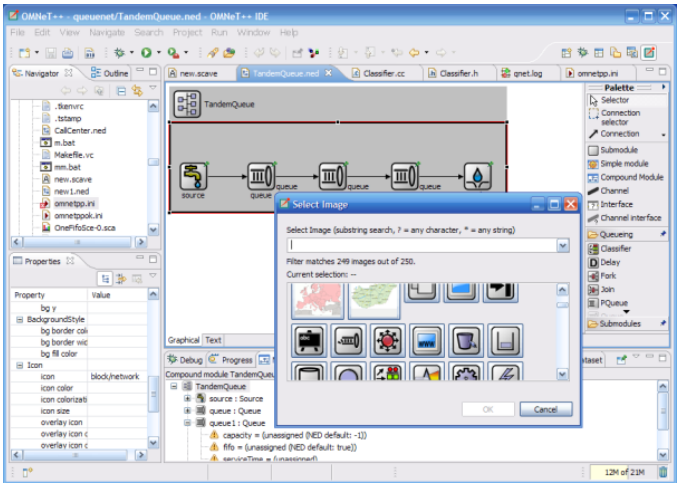
Nesta seção serão descritas as ferramentas que serão utilizadas no desenvolvimento do projeto, bem como suas finalidades e objetivos para a pesquisa.

3.2.1 OMNeT++

O OMNeT++ é um simulador estruturado em C++ baseado em componentes amplamente utilizado para modelar sistemas de comunicação, redes de computadores, sistemas multiprocessadores e outros sistemas distribuídos. Nesse sentido, o conceito de rede é extenso e inclui redes de comunicação com e sem fio, redes *on-chip*, redes de filas etc. (OpenSim Ltd., 2019).

O simulador fornece uma arquitetura modular que permite que os pesquisadores construam modelos de rede complexos usando componentes de software reutilizáveis, chamados módulos. Esses módulos podem ser configurados para representar nós de rede, links de comunicação, protocolos de rede e outros elementos de um sistema de comunicação (Figura 10). Isso permite uma representação precisa e detalhada dos protocolos de comunicação em estudo (OpenSim Ltd., 2019).

Figura 10 - Captura de tela do ambiente de desenvolvimento do simulador OMNET++.



Fonte: OpenSim Ltd., 2019.

As principais funcionalidades do OMNeT++ incluem funções específicas de domínio, como suporte para redes de sensores, redes ad-hoc sem fio, protocolos de comunicação, modelagem de desempenho. Além disso, o simulador oferece um ambiente de desenvolvimento com linguagem de programação, um ambiente de tempo de execução gráfico e uma série de outras ferramentas. Existem extensões para simulação em tempo real, emulação de rede, integração de banco de dados, integração SystemC e diversas outras (OpenSim Ltd., 2019).

Em suma, o OMNeT++ oferece uma plataforma abrangente e flexível para a simulação e análise de protocolos de comunicação e estratégias de eficiência de comunicação.

3.3 Etapas metodológicas

A seguir serão descritas as etapas metodológicas do projeto, sendo elas: modelo de simulação, definição de cenários de testes, critérios de avaliação e simulação dos cenários de testes.

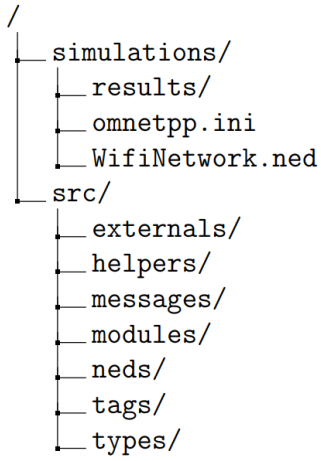
3.3.1 Modelo de simulação

Tomando como base Gumus (2024), o modelo experimental utilizado no trabalho foi reproduzido em ambiente simulado através do software OMNET++. Essa ferramenta é amplamente utilizada para simulações de rede de computadores e contém suporte para mudança e aplicação de protocolos de comunicação.

3.3.1.1 Organização do projeto

Dentro do simulador, o projeto é organizado como um diretório de pastas (Figura 11), onde cada pasta realiza uma função que auxilia a simulação a se desenvolver.

Figura 11 - Organização do projeto no simulador.



Fonte: Gumus, 2024.

Desta forma, o projeto possui dois principais subpastas: “simulations” e “src”. A pasta “simulations” guarda os arquivos principais de descrição da simulação, como arquivos de definição de rede (WifiNetwork.ned) e arquivos de inicialização (omnetpp.ini), além de poder guardar os resultados da simulação (results), caso necessário. Cada arquivo se define como:

a) WifiNetwork.ned: arquivo NED utilizado para construir e modificar a topologia de rede;
b) omnetpp.ini: arquivo de configuração utilizado para customização de parâmetros e módulos de simulação;
c) results: pasta para resultados gerados pela simulação, de forma escalar e vetorial.

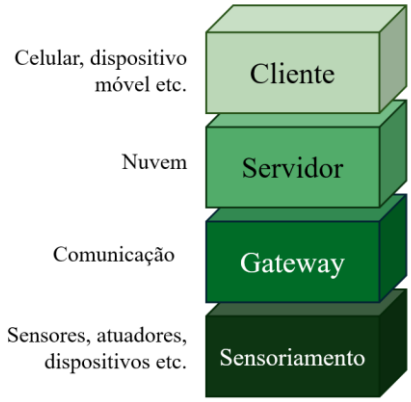
Já a pasta “src” contém todos os arquivos fonte que definem e geram o comportamento da simulação, a maioria dos arquivos é escrita em C++, com exceção dos arquivos de topologia de rede que são escritos na linguagem NED. Na pasta “src” ainda é possível ver outros três diretórios, sendo eles: “neds”, “modules” e “messages”. No diretório “neds” são inseridos os arquivos do tipo .ned, ou seja, os arquivos que descrevem a topologia e o comportamento de rede que, no caso dessa pesquisa, servem para reproduzir o comportamento do MQTT-SN. No diretório “modules” são definidos os arquivos em C++ que representam as classes que auxiliam na implementação do MQTT-SN. Por fim, na seção “modules”, são definidas as mensagens que o MQTT-SN utiliza na definição dos seus módulos, através dos arquivos fonte escritos em C++. As pastas remanescentes contêm arquivos auxiliares e outras utilidades. Os parâmetros e classes

definidos nesta pasta secundária podem ser alterados conforme o objetivo nos arquivos de inicialização (.ini) (Gumus, 2024).

3.3.1.2 Topologia de rede

Uma rede com dispositivos IoT segue uma topologia, conforme a Figura 12, onde a base são os sensores, que enviarão informações através de uma porta de entrada de comunicação (*gateway)* para o servidor que posteriormente compartilha essa informação com um dispositivo móvel (cliente), se desejado (BASSI, EUROPE e HORN, 2008).

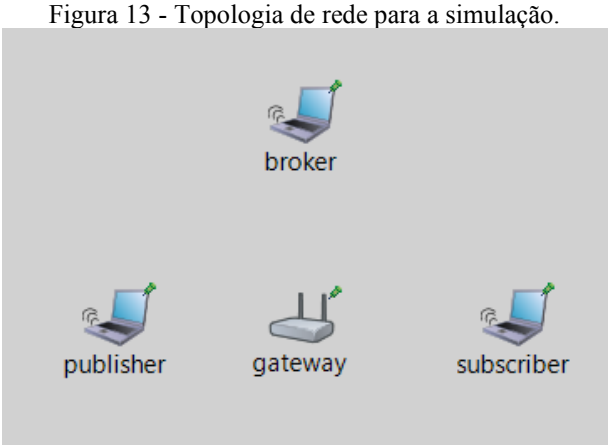
Figura 12 - Estrutura básica de uma arquitetura IoT.



Fonte: Autoria Própria.

A rede simulada nesse projeto se baseia em uma rede básica de comunicação sem fio onde todos os dispositivos utilizam o padrão IEEE 802.11 (Wi-Fi), de forma geral, contendo um publicador (*publisher),* um assinante (*subscriber)*, um *gateway* e um *broker* (Figura 13), possibilitando variações de quantidade de acordo com os cenários a serem simulados, onde os publicadores enviam dados para os assinantes através de um *broker*.

Figura 13 - Topologia de rede para a simulação.

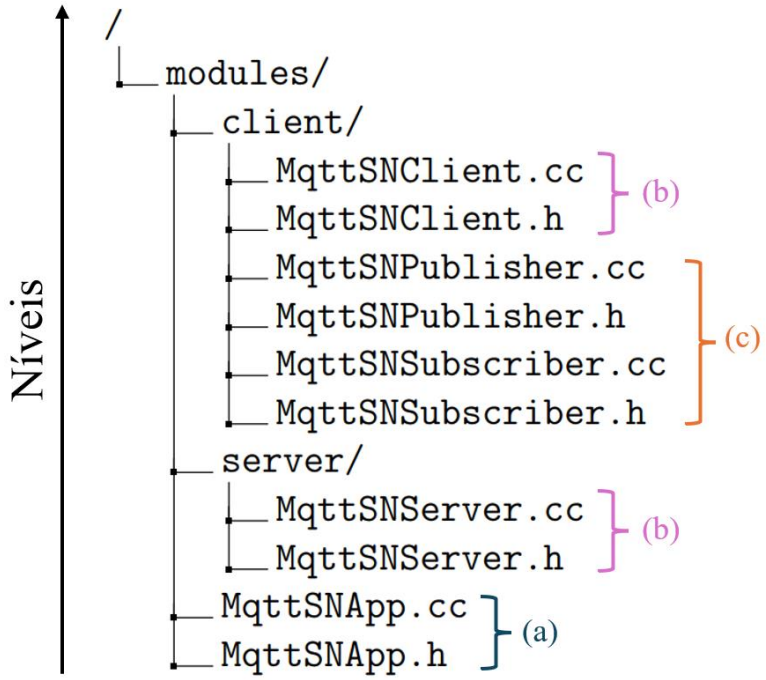


Fonte: Gumus, 2024. (Adaptado)

3.3.1.3 Classes

As classes criadas para a simulação estão alocadas no diretório “modules” citado anteriormente, onde estão organizadas (Figura 14) de tal forma que se veja a hierarquia previamente definida pelo próprio protocolo MQTT-SN.

Figura 14 - Organização das classes no diretório modules.



Fonte: Gumus, 2024 (Adaptado).

No primeiro nível, está definida a classe MqttSNApp (Figura 14a), que descreve os atributos e métodos gerais que serão herdados pelas classes superiores, como a comunicação utilizando UDP e a atuação na camada de aplicação através do framework INET, que é utilizado ao longo de todas as subclasses para enviar e receber pacotes de informações através da rede, na camada de transporte. Em seguida, as classes MqttSNClient e MqttSNServer são derivadas da classe anterior, conforme a Figura 14b. A classe MqttSNClient implementa todas as funcionalidades necessárias para uma operação do cliente MQTT-SN, incluindo a conexão com o gateway, mecanismo de retransmissão para mensagens e gerenciamento de transição de estado. Já a classe MqttSNServer desempenha um papel central no funcionamento do protocolo, aplicando as funcionalidades descritas em suas especificações. Por conseguinte, as classes MqttSNPublisher e MqttSNSubscriber (Figura 14c) herdam as configurações das classes anteriores, funções especificas para o seu papel, como procedimentos de registro de tópico e publicação de mensagem e habilitação de assinatura de tópico, cancelamento de assinatura e gerenciamento mensagens de publicação recebidas do *broker* (Gumus, 2024).

3.3.1.4 Parâmetros de inicialização

Durante a criação de uma simulação dentro do OMNeT++, um aspecto importante a ser considerado são os parâmetros de inicialização. Nesse sentido, o arquivo omnetpp.ini concentra todas as definições para customização e configuração de cada recurso de simulação. Todos os cenários definidos para serem testados nessa pesquisa serão configurados através desse arquivo.

Em primeiro lugar, foi definida a configuração de conexão entre os dispositivos. Nesse sentido, foram atribuídos endereços de IP manuais (Figura 15) para diminuir o processamento da simulação. Na camada de aplicação, cada nó deve vincular seu soquete UDP com seu próprio endereço IP e porta local, que também é especificado no arquivo de configuração.

Figura 15 - Configuração de IP no arquivo de inicialização omnetpp.ini.

```
#=========== Configuração de IP ==============
*.configurator.config = xml("<config> \
                              <interface hosts='server' address='10.0.0.5'/>\
                              <interface hosts='publisher1' address='10.0.0.20'/>\
                              <interface hosts='subscriber1' address='10.0.0.21'/>\
                            </config>")

*.*.ipv4.ip.limitedBroadcast = true
```

Fonte: Gumus, 2024 (Adaptado).

Em seguida, as configurações relacionadas ao cliente, como conexão com o gateway, intervalos de publicação e inscrição (Figura 16), e tempo de simulação, que também foram definidos nessa etapa. Os parâmetros mais utilizados e variáveis são aqueles relacionados aos publicadores e assinantes.

Figura 16 - Configuração para os *publishers* e *subscribers*, com seus tempos de publicação e inscrição.

```
#=========== Configuração do Publisher ==============

*.publisher*.numApps = 1
*.publisher*.app[0].typename = "MqttSNPublisher"
*.publisher*.app[0].localPort = 1000
*.publisher*.app[0].destPort = 1000

*.publisher*.app[0].publishMinusOneDestAddress = "server"
*.publisher*.app[0].publishMinusOneDestPort = 1000

*.publisher*.app[0].activeStateInterval = -1s
*.publisher*.app[0].publishInterval = 5s #intervalo de publicação
#=========== Configuração do Subscriber ==============
*.subscriber*.numApps = 1
*.subscriber*.app[0].typename = "MqttSNSubscriber"
*.subscriber*.app[0].localPort = 1000
*.subscriber*.app[0].destPort = 1000

*.subscriber*.app[0].activeStateInterval = -1s
*.subscriber*.app[0].subscriptionInterval = 2s #intervalo de inscrição
```

Fonte: Gumus, 2024 (Adaptado).

Em particular, ainda no arquivo de parâmetros de inicialização, foram definidos valores formatados em JSON indicando os tópicos a serem disponibilizados pelos publicadores e seus

dados a serem publicados durante a simulação. Esse JSON possui o tamanho de 44 bytes e será utilizado para a análise de resultados posteriormente.

O número de registros e publicações e seus intervalos associados estão entre os parâmetros definidos. Da mesma forma, valores formatados em JSON são usados para assinantes, especificando os tópicos a serem assinados (Figura 17). Operações de assinatura e cancelamento de assinatura também são parametrizadas.

Figura 17 - JSON com as informações a serem publicadas e recebidas pelos clientes.

```
#=========== Definição dos tópicos a serem inscritos ==============

*.subscriber1.app[0].itemsJson = "[\
    { \"topic\": \"temperature\", \"idType\": \"normal\", \"qos\": 2 },\
    { \"topic\": \"humidity\", \"idType\": \"normal\", \"qos\": 2 }\
]"

#=========== Tópicos e informações a serem publicadas ==============

*.publisher1.app[0].itemsJson = "[\
    {\
        \"topic\": \"temperature\",\
        \"idType\": \"normal\",\
        \"data\": [\
            { \"qos\": 1, \"retain\": false, \"data\": \"TemperatureDataA\" },\
            { \"qos\": 1, \"retain\": false, \"data\": \"TemperatureDataB\" },\
            { \"qos\": 1, \"retain\": false, \"data\": \"TemperatureDataC\" }\
        ]\
    },\
    {\
        \"topic\": \"humidity\",\
        \"idType\": \"normal\",\
        \"data\": [\
            { \"qos\": 1, \"retain\": false, \"data\": \"HumidityDataA\" },\
            { \"qos\": 1, \"retain\": false, \"data\": \"HumidityDataB\" }\
        ]\
    }\
]"
```

Fonte: Gumus, 2024 (Adaptado).

Além disso, os níveis de Qualidade de Serviço, cruciais para a análise geral, são especificados como campos dentro das descrições JSON.

3.3.1.5 Impressão de resultados

Dentro dos módulos definidos nas seções citadas anteriormente, definiu-se alguns valores que deverão ser impressos ao fim de cada simulação, sendo eles: quantidade de mensagens enviadas e recebidas, latência (*End-to-End Delay*) e eficácia (*Hit Rage*) (Gumus, 2024).

A latência é definida com base no parâmetro "*End-to-End Delay"* que identifica o atraso de pacotes publicados *versus* os pacotes recebidos, conforme a equação 1:

$$Latência = \frac{\sum_{i=1}^{N}(T_{chegada,i} - T_{envio,i})}{N} \quad (1)$$

onde N representa o número total de publicações recebidas pelos assinantes e T o tempo em segundos.

Já a eficácia da comunicação define-se como:

$$Eficácia = \frac{mensagens\ recebidas}{mensagens\ enviadas} \times 100\% \quad (2)$$

Os resultados dessas métricas podem ser visualizados no console do simulador, conforme a Figura 18.

Figura 18 - Exemplo de impressão de resultados da simulação.

```
==== Publish Messages Results ====
Unique sent: 31
Unique received: 29
Total received: 55
Total received duplicates: 4

Average end-to-end delay: 4.26601 seconds
Hit rate: 93.5484 %
```

Fonte: Autoria Própria.

3.3.2 Definição de cenários de testes

Para avaliar o desempenho do protocolo MQTT-SN em diferentes condições de rede, foram definidos diversos cenários de testes que variam em termos de frequência de publicação, quantidade de dispositivos, tamanho dos dados e qualidade de serviço (QoS). Cada cenário foi projetado para explorar aspectos específicos do comportamento do protocolo, identificando suas limitações e vantagens em diferentes situações. A seguir, são detalhados os cenários de testes considerados nesta pesquisa.

3.3.2.1 Alta frequência de publicação

O primeiro cenário definido se baseia na redução do tempo dos *publishers* para publicação em intervalos menores, assim, Gumus (2024) define o tempo padrão para publicação de 15 segundos, porém, para este cenário, foi considerado o tempo de 0,1 segundos (100 ms), conforme a Figura 19.

Figura 19 - Mudança no valor de intervalo de publicação para o cenário de alta frequência de publicação.

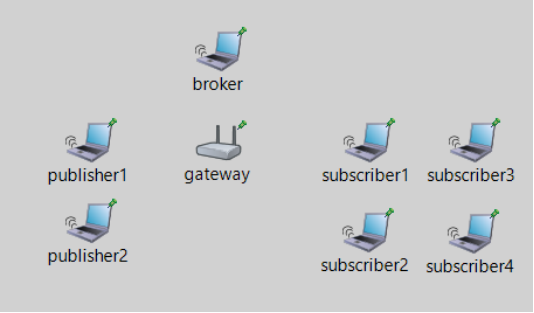
```
*.publisher*.app[0].activeStateInterval = -1s
*.publisher*.app[0].publishInterval = 15s #intervalo de publicação
```

```
*.publisher*.app[0].activeStateInterval = -1s
*.publisher*.app[0].publishInterval = 0.1s #intervalo de publicação
```

Fonte: Gumus, 2024 (Adaptado).

Além disso, foi considerado a topologia de rede original (Figura 20), onde se tem dois (2) *publishers* e quatro (4) *subscribers* e o tempo de inscrição padrão, definido para 2 segundos.

Figura 20 - Topologia de rede para o cenário de alta frequência de publicação.



Fonte: Gumus, 2024 (Adaptado).

A Tabela 1 resume todas as características do cenário descrito.

Tabela 1 - Resumo das características para o cenário de alta frequência de publicação

Configuração	Valor
Publishers	2
Subscribers	4
Tempo de publicação (s)	0,1
Tempo de inscrição (s)	2

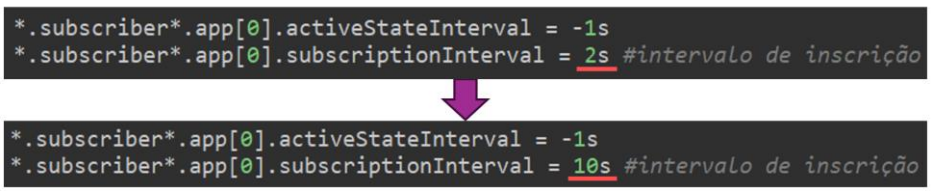
Fonte: Autoria Própria.

O principal objetivo desse cenário é avaliar o comportamento da rede quando as publicações ocorrem com maior frequência. Essa análise permite observar se há degradação no desempenho da rede devido ao aumento da frequência de mensagens.

3.3.2.2 Assinatura atrasada

Em um cenário de assinatura atrasada, o tempo de inscrição dos *subscribers* é grande quando comparado ao tempo de publicação e, para acompanhar essa definição, o tempo de inscrição foi definido para acontecer a cada 10 segundos (Figura 21). A topologia de rede se mantém a mesma utilizado para o cenário de alta frequência de publicação (Figura 20).

Figura 21 - Mudança no valor de intervalo de inscrição para o cenário de assinatura atrasada.



Fonte: Gumus, 2024 (Adaptado).

Em relação a outros parâmetros, foram considerados os valores já definidos posteriormente, com exceção do tempo de publicação, que foi definido para 5 segundos, conforme a Tabela 2.

Tabela 2 - Resumo das características para o cenário de assinatura atrasada

Configuração	Valor
Publishers	2
Subscribers	4
Tempo de publicação (s)	5
Tempo de inscrição (s)	10

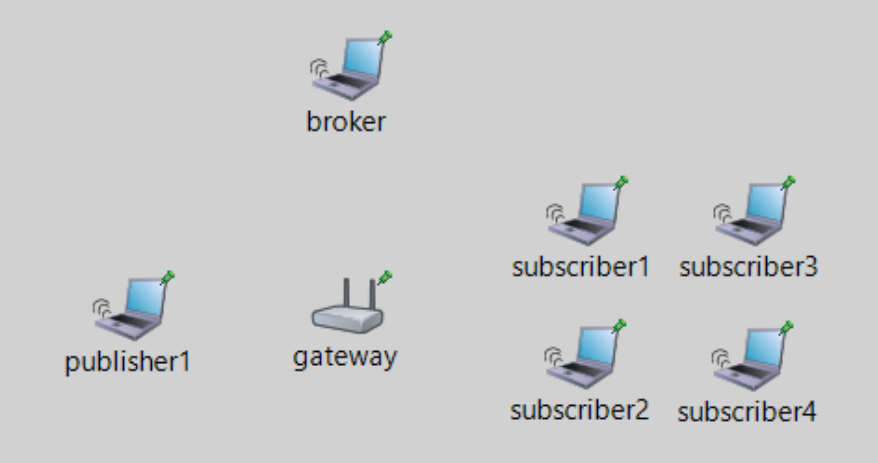
Fonte: Autoria Própria.

Ao simular esse resultado, será possível avaliar o impacto da inscrição tardia na taxa de entrega de mensagens e na latência. Esse cenário auxilia a entender o comportamento da rede quando os assinantes não estão prontos no início da comunicação e entram em operação posteriormente.

3.3.2.3 Broadcast

Broadcast de rede é definido como a transmissão de informações para múltiplos destinatários simultaneamente. Dessa maneira, para avaliar o desempenho de rede, foi definida uma topologia de rede onde existe somente um *publisher* na rede, enviando informações para vários *subscribers* (Figura 22).

Figura 22 - Topologia de rede para o cenário de *broadcast*.



Fonte: Gumus, 2024 (Adaptado).

Os demais parâmetros não foram alterados de acordo com o padrão definido anteriormente, conforme a Tabela 3.

Tabela 3 - Resumo das características para o cenário de *broadcast*

Configuração	Valor
Publishers	1
Subscribers	4
Tempo de publicação (s)	15
Tempo de inscrição (s)	10

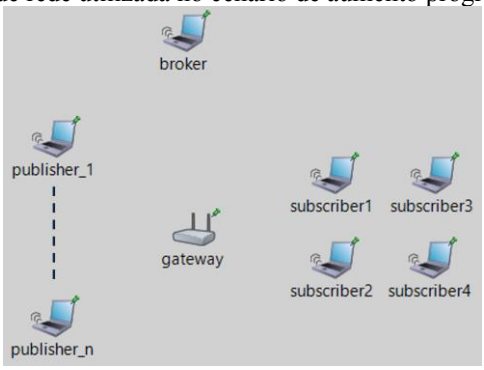
Fonte: Autoria Própria.

O objetivo principal desse cenário é analisar a capacidade do protocolo em distribuir dados para múltiplos destinatários e identificar os impactos no desempenho.

3.3.2.4 Aumento progressivo de publicadores

Uma outra possibilidade de avaliar o desempenho do protocolo e da rede é aumentar progressivamente os clientes, sendo eles publicadores ou assinantes. No caso dessa pesquisa, foi escolhido variar o número *n* de publicadores na rede, aumento-o progressivamente. Para isso, a cada novo número de publicadores, a topologia de rede é ajustada, assim como seus valores de IP respectivos, de acordo com a Figura 23.

Figura 23 - Topologia de rede utilizada no cenário de aumento progressivo de publicadores.



Fonte: Gumus, 2024 (Adaptado).

O número de assinantes foi mantido, alterando somente os intervalos de publicação e inscrição, de acordo com a Tabela 4.

Tabela 4 - Resumo das características para o cenário de aumento progressivo de publicadores

Item	Valor
Publishers	*n*
Subscribers	4
Tempo de publicação (s)	5
Tempo de inscrição (s)	2

Fonte: Autoria Própria.

Dessa forma, será possível avaliar como o aumento do número de publicadores afeta o desempenho da rede, incluindo a latência, a taxa de entrega de mensagens e a sobrecarga da rede. Esse cenário ajuda a entender a escalabilidade do protocolo em um ambiente com maior carga de publicação.

3.3.2.5 Variação na qualidade de serviço (QoS)

Nos *JavaScript Object Notation* (JSON) definidos nos parâmetros de inicialização também são definidos níveis de QoS, que podem ser variados para analisar o comportamento da rede. Nesse sentido, a topologia de rede foi fixada conforme a Figura 20, onde se tem dois (2) publicadores e quatro (4) assinantes. Além disso, os intervalos de publicação e inscrição foram fixados conforme o cenário anterior, em 5 e 2 segundos, respectivamente. A Tabela 5 resume os parâmetros desse cenário, onde *n* representa os níveis de QoS.

Tabela 5 - Resumo das características para o cenário de variação de QoS.

Item	**Valor**
Publishers	2
Subscribers	4
Tempo de publicação (s)	5
Tempo de inscrição (s)	2
Quality of Service (QoS)	*n*

Fonte: Autoria Própria.

O objetivo desse cenário é avaliar o impacto dos diferentes níveis de QoS na latência e na confiabilidade da entrega de mensagens. Esse cenário auxilia na definição de qual nível de QoS é mais adequado para diferentes tipos de aplicação IoT.

3.3.3 Critérios de avaliação

O principal objetivo desse estudo é realizar uma análise de desempenho do protocolo MQTT-SN em diversos cenários. Para isso, serão estabelecidos critérios para verificação após a execução das simulações, com base em (Cosmi e Mota, 2019).

Além disso, será feito o uso de recursos visuais como gráficos e tabelas para aprofundar o status e obter os resultados mais detalhados com possibilidade de análise sucinta e simplificada. Essa análise será feita nas seções a seguir com base nos dados coletados durante as simulações. Serão analisados dados tangíveis de tempo de transmissão, latência, *throughput*, taxa de entrega de mensagens e *overhead,* de acordo com o Quadro 1.

Quadro 1 - Métricas a serem avaliadas na pesquisa, sua descrição e unidades de medida.

Métrica	Descrição	Unidade de Medida
Latência	Atraso na comunicação	Segundos (s)
Throughput	Taxa em que os dados são transmitidos com sucesso	Bits por segundo (bps)
Taxa de entrega de mensagens	Mensagens recebidas em relação às mensagens enviadas	Porcentagem (%)
Overhead	Quantidade de dados perdidos ao longo do tempo	Quantidade de mensagens duplicadas

Fonte: Autoria Própria.

Com base nos dados impressos ao fim da simulação, ainda, será possível calcular o *Throughput* de acordo com a equação*:*

$$Throughput = \frac{tamanho\ da\ mensagem \times nº\ mensagens\ enviadas}{tempo\ de\ simulação} \quad (3)$$

Como a mensagem se manterá fixa no valor de 44 bytes, tem-se:

$$Throughput = \frac{44\ bytes \times nº\ mensagens\ enviadas}{tempo\ de\ simulação} \quad (4)$$

Para isso, serão realizadas determinadas execuções de simulações a fim de obter um espectro amostral suficiente para embasar os resultados.

3.3.4 Simulação dos cenários de testes

Nesta seção será descrito o passo a passo para a realização da simulação dos cenários de testes definidos anteriormente, cada cenário possui objetivos e configurações específicas que deverão ser adaptadas para o seu devido fim.

3.3.4.1 Simulação do cenário de alta frequência de publicação

Em primeiro lugar, foram realizados os ajustes conforme a tabela de parâmetros do cenário (Tabela 1), principalmente a redução do intervalo de publicação (Figura 24).

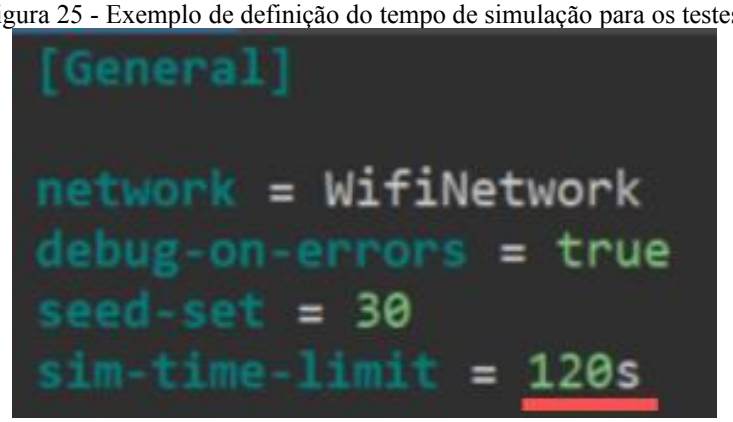
Figura 24 - Definição do intervalo de publicação para 100 ms.

```
*.publisher*.app[0].activeStateInterval = -1s
*.publisher*.app[0].publishInterval = 0.1s #intervalo de publicação
```

Fonte: Autoria Própria.

Em seguida, o tempo de simulação foi inserido no arquivo de configuração de parâmetros iniciais da simulação (omnetpp.ini), para auxiliar na coleta de resultados. Nesse cenário, os parâmetros da tabela foram mantidos fixos enquanto os valores de simulação foram alterados para: 30, 45, 60, 90 e 120 segundos (Figura 25).

Figura 25 - Exemplo de definição do tempo de simulação para os testes.

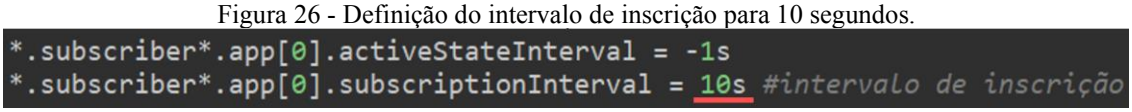


Fonte: Autoria Própria.

3.3.4.2 Simulação do cenário de assinatura atrasada

O primeiro ajuste realizado se refere aos parâmetros definidos na tabela na seção de cenários, conforme a Tabela 2, onde o intervalo de inscrição foi configurado para 10 segundos.

Figura 26 - Definição do intervalo de inscrição para 10 segundos.



Fonte: Autoria Própria.

Da mesma forma realizada no cenário anterior, também foi variado o tempo de simulação, para a coleta de dados.

3.3.4.3 Simulação do cenário de *broadcast*

Além das atualizações de parâmetros de acordo com a tabela de características do cenário a ser simulado, para o cenário de *broadcast*, também foram atualizados o JSON de dados e os endereços de IPs, retirando o *publisher2*. Nesse cenário também foi variado o tempo de simulação, na escala de 30, 45, 60, 90 e 120 segundos.

3.3.4.4 Simulação do cenário de aumento progressivo de publicadores

Diferentemente dos cenários anteriores, as variações nesse cenário foram feitas em relação a quantidade de publicadores, além dos parâmetros característicos. Os números de publicadores foram definidos em 2, 10, 15, 20 e 30. Para cada novo publicador, foi necessário incluir mais uma cópia do JSON de tópicos e um novo endereço de IP, conforme a Tabela 6, onde n representa o número do publicador e x os dois algarismos finais do endereço de IP.

Tabela 6 - Endereçamento de IP feito para o cenário de aumento de publicadores.

Publicador	Endereço de IP
publisher1	10.0.0.20
publisher2	10.0.0.21
publisher3	10.0.0.22
publisher4	10.0.0.23
publisher_n	10.0.0.(x+1)

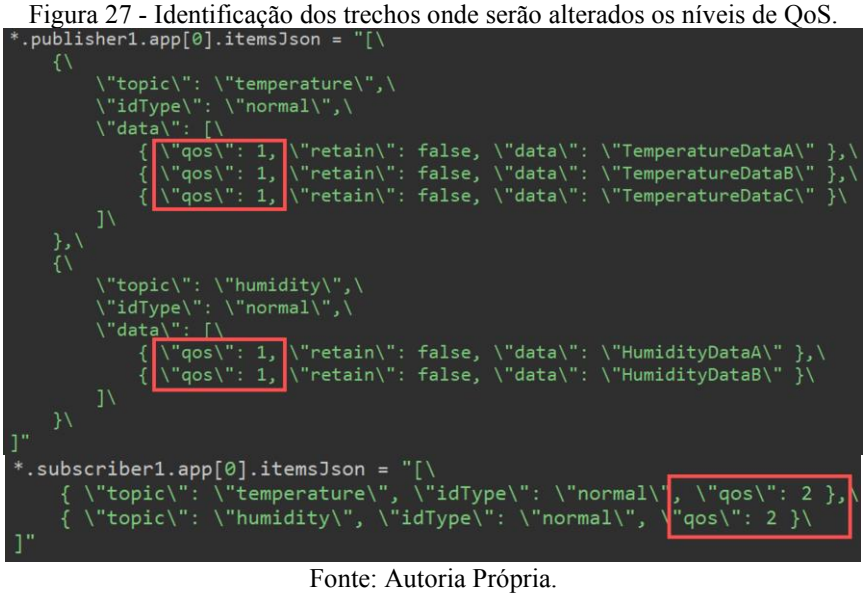
Fonte: Autoria Própria.

Além disso, a cada nova adição de publicador é necessário atualizar o arquivo de topologia de rede.

3.3.4.5 Simulação do cenário de variação de QoS

A variação de QoS também compõe um cenário único de simulação, onde, após as configurações iniciais de acordo com a tabela características, são realizadas somente alterações no JSON de publicação e inscrição, para os valores de QoS.

Figura 27 - Identificação dos trechos onde serão alterados os níveis de QoS.



Fonte: Autoria Própria.

3.4 Limitações

Outro ponto a ser considerado na metodologia são as limitações que podem impactar os resultados do estudo. Em primeiro lugar, foi analisado apenas um protocolo de comunicação e com isso o estudo pode não abranger todas as possibilidades de aplicações. Ainda, como o estudo será reproduzido em ambiente simulado e impactos como *hardware* não poderão ser considerados. Por fim, os protocolos de comunicação estão em constante atualizações e este estudo pode não ser o mais atual.

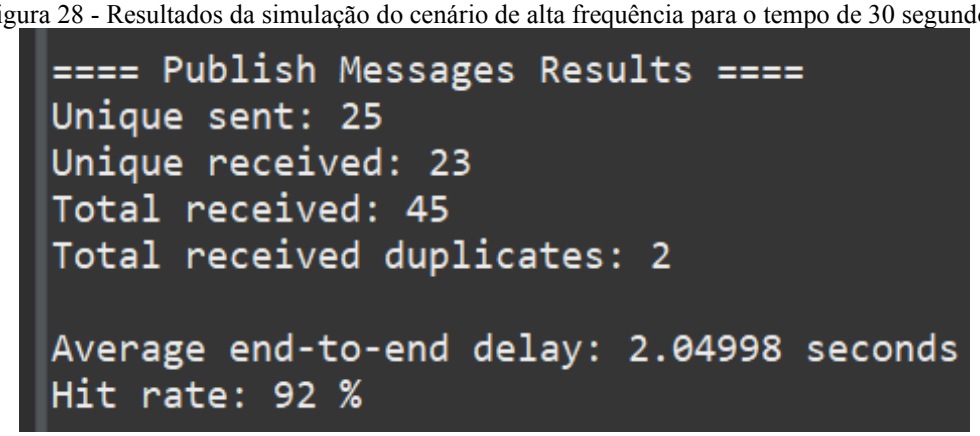
4 RESULTADOS E DISCUSSÕES

Esta seção tem como objetivo demonstrar os resultados obtidos nas simulações através de gráficos e tabelas e assim, realizar inferências acerca deles.

4.1 Resultados do cenário de alta frequência de publicação

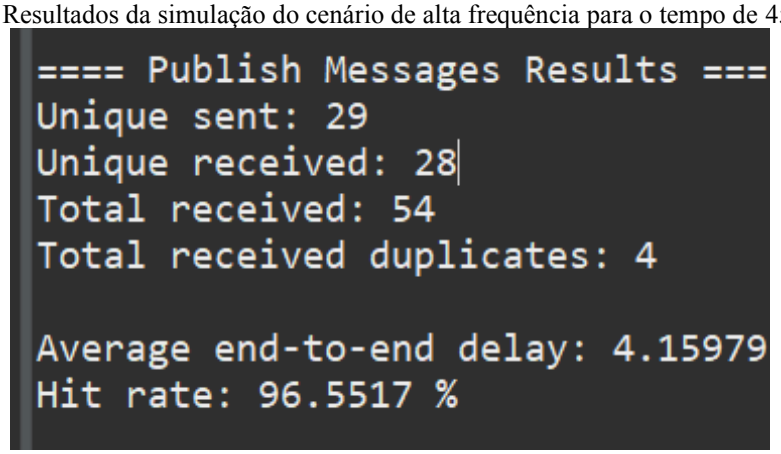
Para o cenário de alta frequência, foi variado o tempo de simulação, a fim de obter diversos valores para auxiliar na visualização do comportamento da rede nesse caso. Para o tempo de simulação inicial, foi considerado 30 segundos, obtendo-se os seguintes resultados ilustrados na Figura 28. Em seguida, o tempo de simulação foi definido para 45 segundos, conforme a Figura 29.

Figura 28 - Resultados da simulação do cenário de alta frequência para o tempo de 30 segundos.



Fonte: Autoria Própria.

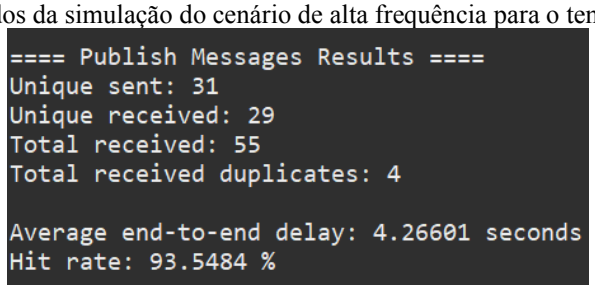
Figura 29 - Resultados da simulação do cenário de alta frequência para o tempo de 45 segundos.



Fonte: Autoria Própria.

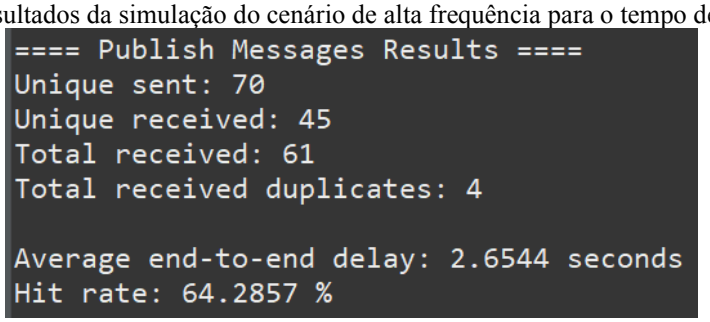
Por fim, o processo foi repetido para os tempos de simulação de 60, 90 e 120 segundos, ilustrados nas Figura 30, Figura 31 e Figura 32, respectivamente.

Figura 30 - Resultados da simulação do cenário de alta frequência para o tempo de 60 segundos.



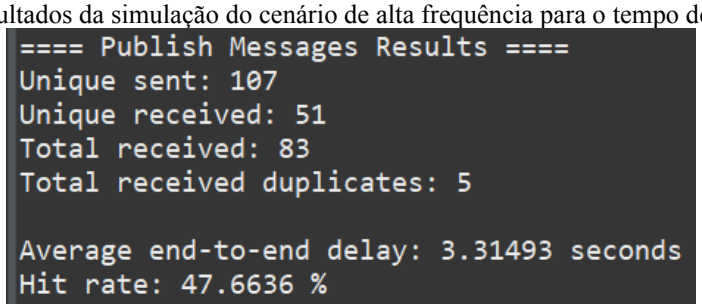
Fonte: Autoria Própria.

Figura 31 - Resultados da simulação do cenário de alta frequência para o tempo de 90 segundos.



Fonte: Autoria Própria.

Figura 32 - Resultados da simulação do cenário de alta frequência para o tempo de 120 segundos.



Fonte: Autoria Própria.

Com base nos resultados impressos, utilizando a equação 4, foi possível definir os valores de *throughput* e elaborar o Quadro 2 com esses resultados, definido a seguir.

Quadro 2 - Resumo dos resultados obtidos na simulação do cenário de alta frequência.

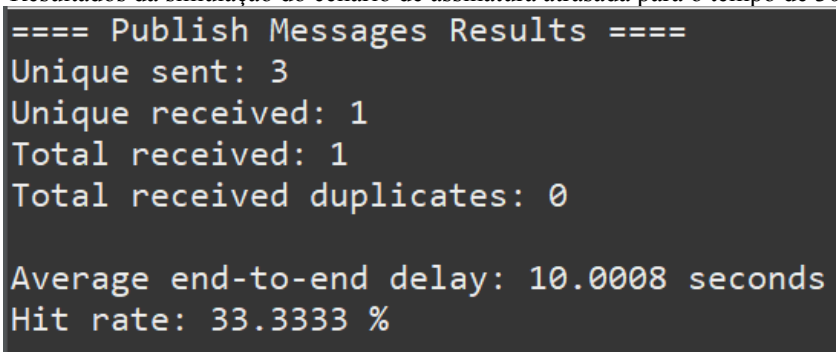
	Tempo de Transmissão (s)				
Métrica	**30**	**45**	**60**	**90**	**120**
Mensagens enviadas	25	29	31	70	107
Mensagens recebidas	23	28	29	45	51
Mensagens duplicadas	2	4	4	4	5
Latência (s)	2,050	4,160	4,266	2,654	3,315
Eficácia (%)	92,00	96,55	93,55	64,29	47,66
***Throughput* (bps)**	36,67	28,36	22,73	34,22	39,23

Fonte: Autoria Própria.

4.2 Resultados do cenário de assinatura atrasada

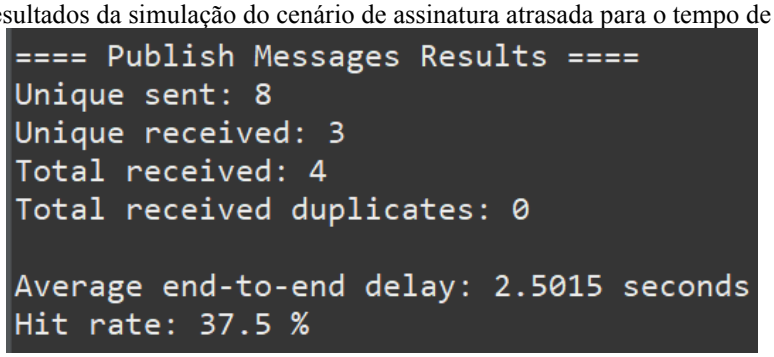
De forma semelhante, para o cenário onde o intervalo de assinatura é elevado, o tempo de simulação foi variado no intervalo de 30 a 120 segundos. Inicialmente, o tempo de simulação foi definido como 30 segundos, em seguida, foram observados e coletados os resultados ilustrados na Figura 33. O procedimento foi reproduzido para os tempos de simulação de 45, 60, 90 e 120 segundos, ilustrados nas Figura 34, Figura 35, Figura 36 e Figura 37, respectivamente. A partir dos resultados obtidos, foram calculados os valores para o *throughput* e o Quadro 3 foi elaborado agrupando todas as informações.

Figura 33 - Resultados da simulação do cenário de assinatura atrasada para o tempo de 30 segundos.



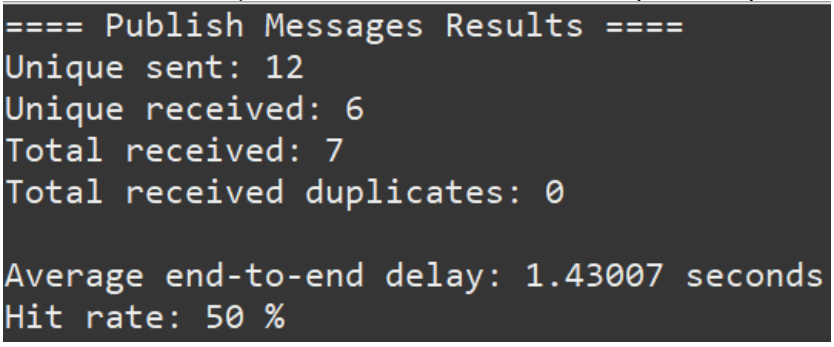
Fonte: Autoria Própria.

Figura 34 - Resultados da simulação do cenário de assinatura atrasada para o tempo de 45 segundos.



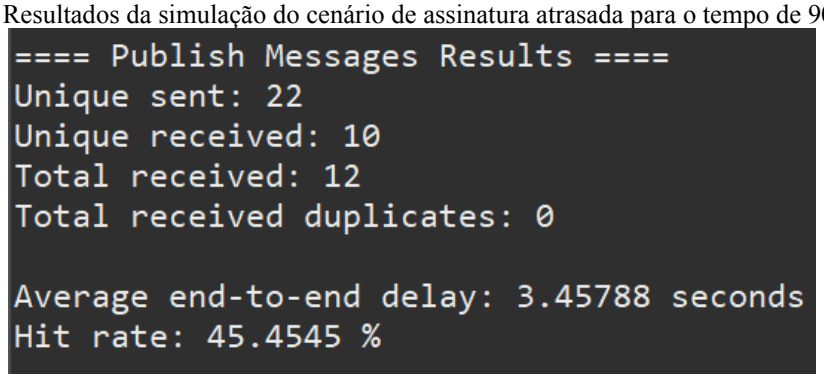
Fonte: Autoria Própria.

Figura 35 - Resultados da simulação do cenário de assinatura atrasada para o tempo de 60 segundos.



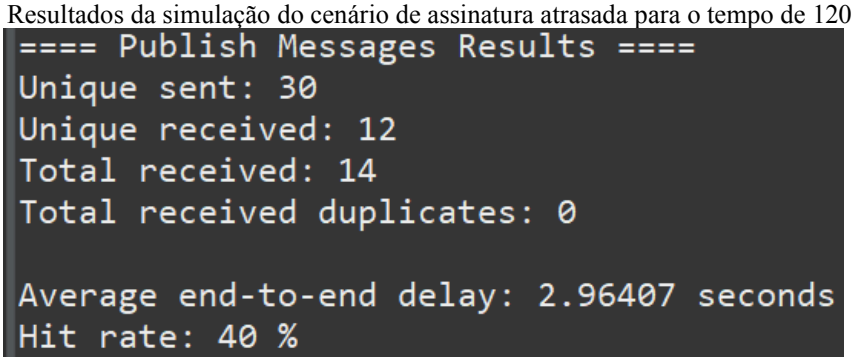
Fonte: Autoria Própria.

Figura 36 - Resultados da simulação do cenário de assinatura atrasada para o tempo de 90 segundos.



Fonte: Autoria Própria.

Figura 37 - Resultados da simulação do cenário de assinatura atrasada para o tempo de 120 segundos.



Fonte: Autoria Própria.

Quadro 3 - Resumo dos resultados obtidos na simulação do cenário de assinatura atrasada.

	Tempo de Transmissão (s)				
Métrica	**30**	**45**	**60**	**90**	**120**
Mensagens enviadas	3	8	12	22	30
Mensagens recebidas	1	3	6	10	12
Mensagens duplicadas	0	0	0	0	0
Latência (s)	10,001	2,502	1,430	3,458	2,964
Eficácia (%)	33,33	37,50	50,00	45,45	40,00
***Throughput* (bps)**	4,40	7,82	8,80	10,76	11,00

Fonte: Autoria Própria.

4.3 Resultados do cenário de *broadcast*

Para o cenário de *broadcast*, que contém apenas um *publisher,* o procedimento de mudança de tempo de simulação foi replicado. Foram coletados os dados conforme ilustrados nas Figura 38, Figura 39, Figura 40, Figura 41 e Figura 42, calculado o valor para o *throughput* em todos os casos e, posteriormente, agrupados no Quadro 4.

Figura 38 - Resultados da simulação do cenário de *broadcast* para o tempo de 30 segundos.

```
==== Publish Messages Results ====
Unique sent: 4
Unique received: 4
Total received: 4
Total received duplicates: 0

Average end-to-end delay: 0.00136401 seconds
Hit rate: 100 %
```

Fonte: Autoria Própria.

Figura 39 - Resultados da simulação do cenário de *broadcast* para o tempo de 45 segundos.

```
==== Publish Messages Results ====
Unique sent: 4
Unique received: 4
Total received: 4
Total received duplicates: 0

Average end-to-end delay: 0.00136401 seconds
Hit rate: 100 %
```

Fonte: Autoria Própria.

Figura 40 - Resultados da simulação do cenário de *broadcast* para o tempo de 60 segundos.

```
==== Publish Messages Results ====
Unique sent: 6
Unique received: 6
Total received: 9
Total received duplicates: 1

Average end-to-end delay: 2.22391 seconds
Hit rate: 100 %
```

Fonte: Autoria Própria.

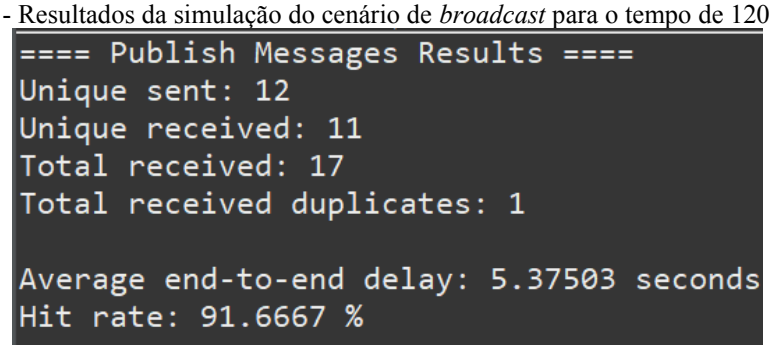
Figura 41 - Resultados da simulação do cenário de *broadcast* para o tempo de 90 segundos.

```
==== Publish Messages Results ====
Unique sent: 10
Unique received: 7
Total received: 10
Total received duplicates: 1

Average end-to-end delay: 4.04866 seconds
Hit rate: 70 %
```

Fonte: Autoria Própria.

Figura 42 - Resultados da simulação do cenário de *broadcast* para o tempo de 120 segundos.



Fonte: Autoria Própria.

Quadro 4 - Resumo dos resultados obtidos na simulação do cenário de *broadcast*.

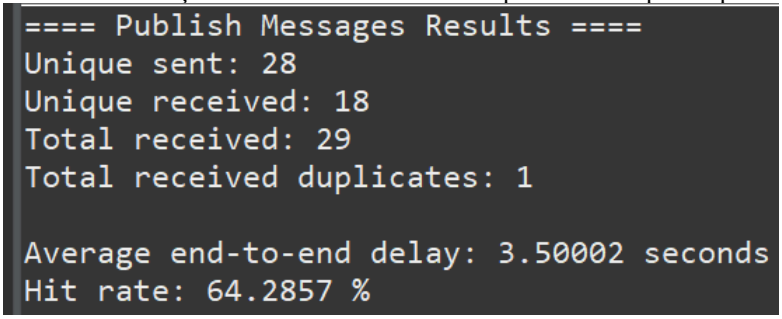
	Tempo de Transmissão (s)				
Métrica	**30**	**45**	**60**	**90**	**120**
Mensagens enviadas	4	4	6	10	12
Mensagens recebidas	4	4	6	7	11
Mensagens duplicadas	0	0	1	1	1
Latência (s)	0,001	0,001	2,224	4,049	5,375
Eficácia (%)	100,00	100,00	100,00	70,00	91,67
***Throughput* (bps)**	5,87	3,91	4,40	4,89	4,40

Fonte: Autoria Própria.

4.4 Resultados do cenário de aumento de publicadores

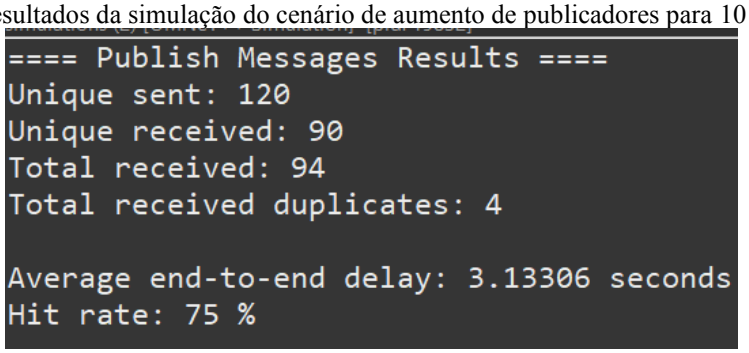
Diferentemente dos cenários anteriores, para o cenário de publicadores aumentando progressivamente, o tempo de simulação foi fixado em 120 segundos. Dessa forma, as Figura 43, Figura 44, Figura 45, Figura 46 e Figura 47 representam os resultados impressos para os casos de 2, 10, 15, 20 e 30 publicadores, respectivamente. Em seguida, foram calculados os valores de *throughput* e os resultados foram compilados no Quadro 5.

Figura 43 - Resultados da simulação do cenário de aumento de publicadores para 2 publicadores.



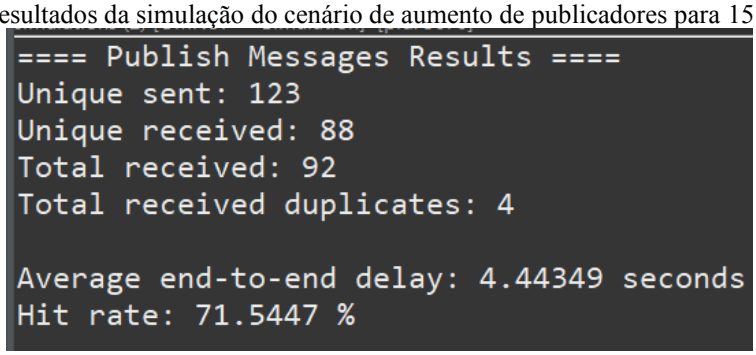
Fonte: Autoria Própria.

Figura 44 - Resultados da simulação do cenário de aumento de publicadores para 10 publicadores.



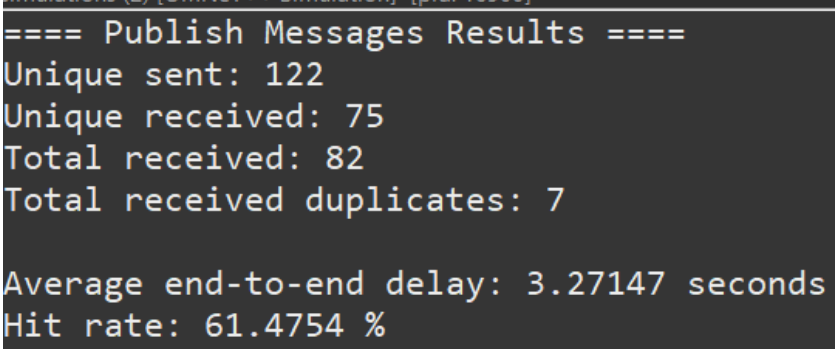
Fonte: Autoria Própria.

Figura 45 - Resultados da simulação do cenário de aumento de publicadores para 15 publicadores.



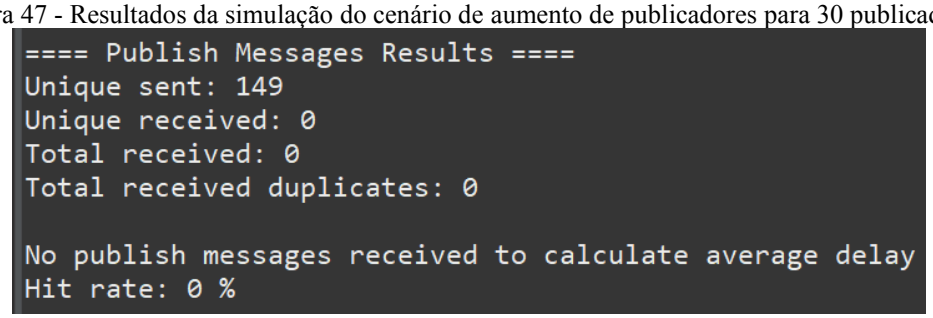
Fonte: Autoria Própria.

Figura 46 - Resultados da simulação do cenário de aumento de publicadores para 20 publicadores.



Fonte: Autoria Própria.

Figura 47 - Resultados da simulação do cenário de aumento de publicadores para 30 publicadores.



Fonte: Autoria Própria.

Quadro 5 - Resumo dos resultados obtidos na simulação do cenário de aumento progressivo de publicadores.

	Quantidade de Publicadores				
Métrica	**2**	**10**	**15**	**20**	**30**
Mensagens enviadas	28	120	123	122	149
Mensagens recebidas	18	90	88	75	0
Mensagens duplicadas	1	4	4	7	0
Latência (s)	3,500	3,133	4,443	3,271	0,000
Eficácia (%)	64,29	75,00	71,54	61,48	0,00
***Throughput* (bps)**	10,27	44,00	45,10	44,73	54,63

Fonte: Autoria Própria.

4.5 Resultados do cenário de variação de QoS

Foram mantidas as configurações características conforme a tabela padrão e os níveis de QoS foram modificados diretamente no JSON de publicação e inscrição. As Figura 48, Figura 49 e Figura 50 ilustram os resultados obtidos para os níveis 0, 1 e 2, respectivamente. No Quadro 6 é possível visualizar todos os dados agrupados, junto dos valores calculados para o *throughput*.

Figura 48 - Resultados da simulação do cenário de variação de QoS para o nível 0.

```
==== Publish Messages Results ====
Unique sent: 40
Unique received: 20
Total received: 37
Total received duplicates: 0

Average end-to-end delay: 0.010862 seconds
Hit rate: 50 %
```

Fonte: Autoria Própria.

Figura 49 - Resultados da simulação do cenário de variação de QoS para o nível 1.

```
==== Publish Messages Results ====
Unique sent: 28
Unique received: 18
Total received: 29
Total received duplicates: 1

Average end-to-end delay: 3.50002 seconds
Hit rate: 64.2857 %
```

Fonte: Autoria Própria.

Figura 50 - Resultados da simulação do cenário de variação de QoS para o nível 2.

```
==== Publish Messages Results ====
Unique sent: 24
Unique received: 21
Total received: 37
Total received duplicates: 0

Average end-to-end delay: 4.13867 seconds
Hit rate: 87.5 %
```

Fonte: Autoria Própria.

Quadro 6 - Resumo dos resultados obtidos na simulação do cenário de variação de QoS.

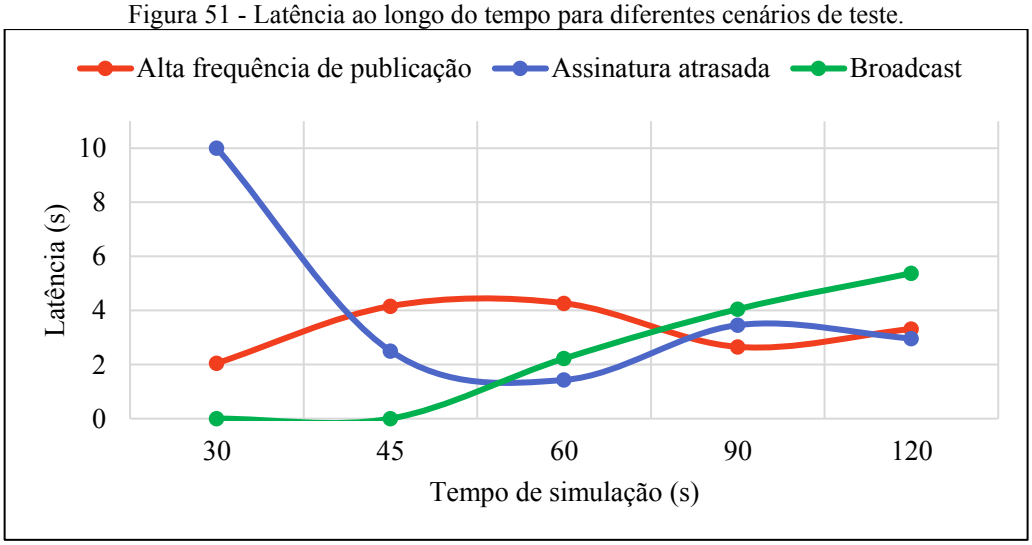
	Nível de QoS		
Métrica	**0**	**1**	**2**
Mensagens enviadas	40	28	24
Mensagens recebidas	20	18	21
Mensagens duplicadas	0	1	0
Latência (s)	0,011	3,500	4,139
Eficácia (%)	50,00	64,29	87,50
***Throughput* (bps)**	14,67	10,27	8,80

Fonte: Autoria Própria.

4.6 Análise de latência, quantidade de mensagens, *throughput*, eficácia e *overhead*

Na Figura 51 estão ilustrados os valores de latência para os cenários de teste onde foi variado o tempo de simulação, na coleta de dados.

Figura 51 - Latência ao longo do tempo para diferentes cenários de teste.

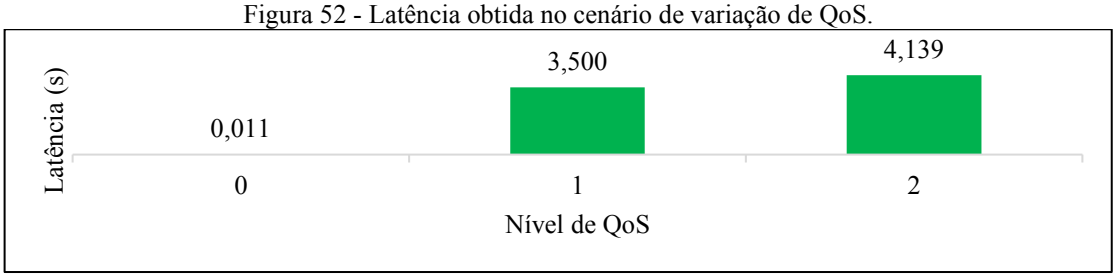


Fonte: Autoria Própria.

Para o cenário de alta frequência de publicação, a latência se mantém entre 2 e 4 segundos, sugerindo que a rede possui capacidade de lidar com uma maior frequência de envio de dados. No cenário de assinatura atrasada, onde os *subscribers* levam mais tempo para se inscrever em um tópico, a latência tende sofre extrema variação com o tempo, além de inicialmente atingir o maior valor medido entre os três cenários, reforçando a dificuldade na sincronização com uma grande nuance entre intervalos de publicação e assinatura. No cenário de *broadcast*, a latência é baixa pela maior parte do tempo, sugerindo que o cenário é mais eficiente para envio de informações, uma vez que carga na rede não aumenta de forma significante.

Realizando a análise para o cenário com diferentes níveis de QoS (Figura 52), observou-se que o nível de QoS = 0 possui a menor latência medida, próxima de 0, que se dá pela definição do nível que não fornece confirmações e garantias de entrega, que resulta em uma menor sobrecarga na rede.

Figura 52 - Latência obtida no cenário de variação de QoS.

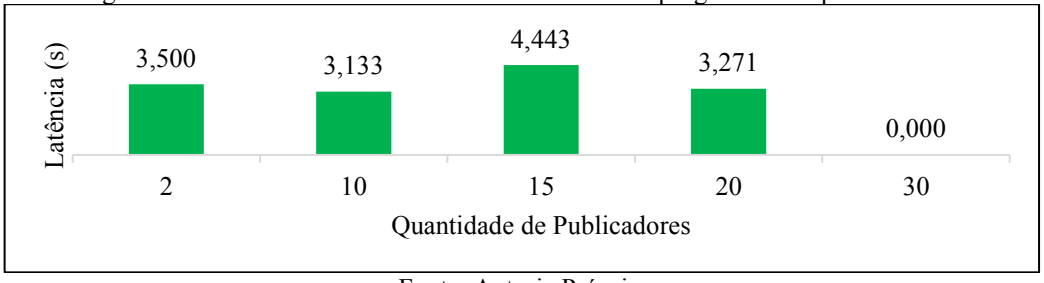


Fonte: Autoria Própria.

Os níveis de QoS 1 e QoS 2 possuem valores próximos e um aumento expressivo quando comparados com o nível de QoS 0, ocorrendo devido ao esforço da rede para entregar mensagens e confirmar seu recebimento e envio.

Ainda, na Figura 53, é possível observar o comportamento da rede no cenário de aumento progressivo de publicadores, que indica um comportamento adaptável da rede na maior parte do tempo, porém, ao aumentar o número de publicadores para 15, a rede sofre uma sobrecarga, atingindo o maior valor de latência.

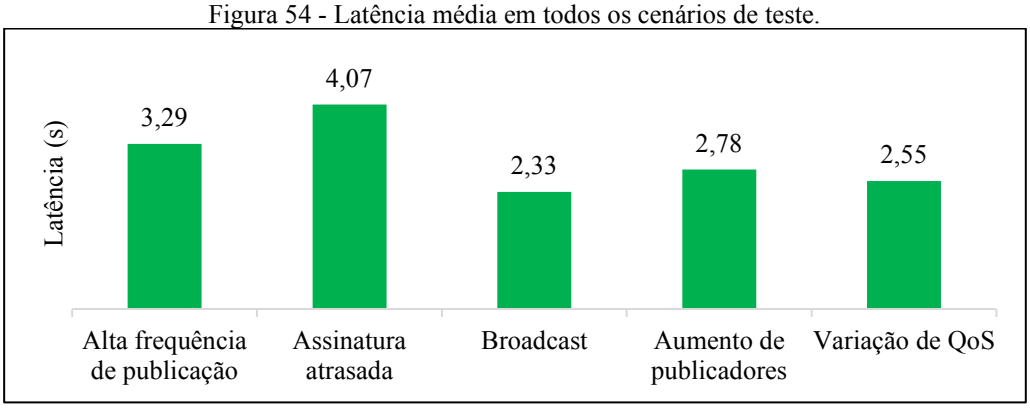
Figura 53 - Latência medida no cenário com aumento progressivo de publicadores.



Fonte: Autoria Própria.

De forma geral, observou-se na Figura 54 que a falta de sincronização entre tempo de inscrição e tempo de publicação garante o maior valor de latência na rede, enquanto os cenários de *broadcast*, aumento de publicadores e variação de QoS se mantém moderados, apesar de ainda serem expressivos. A latência no cenário de alta frequência reforça a ideia de sobrecarga na rede com o alto volume de dados.

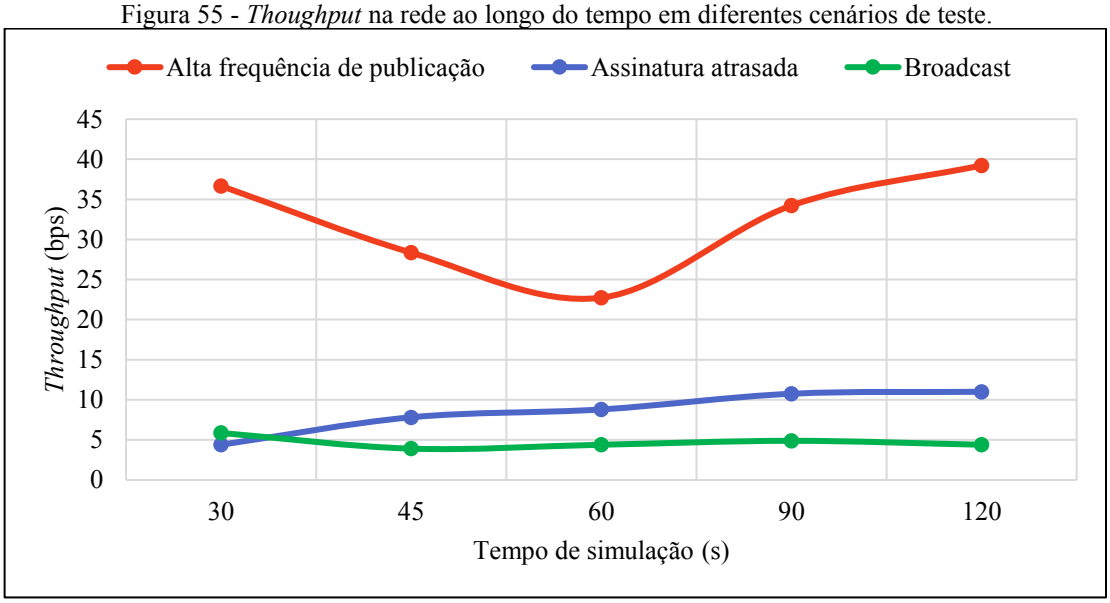
Figura 54 - Latência média em todos os cenários de teste.



Fonte: Autoria Própria.

A Figura 55 mostra como o *throughput* é elevado no cenário de alta frequência de publicação, já que essa métrica é calculada com base nas mensagens enviadas. Em contrapartida, nos outros dois cenários, assinatura atrasada e *broadcast*, o valor de *throughput* é consideravelmente menor, mesmo que o cenário de assinatura atrasada possua mais oscilações de valor que o cenário de *broadcast*, que se demonstrou estável.

Figura 55 - *Thoughput* na rede ao longo do tempo em diferentes cenários de teste.

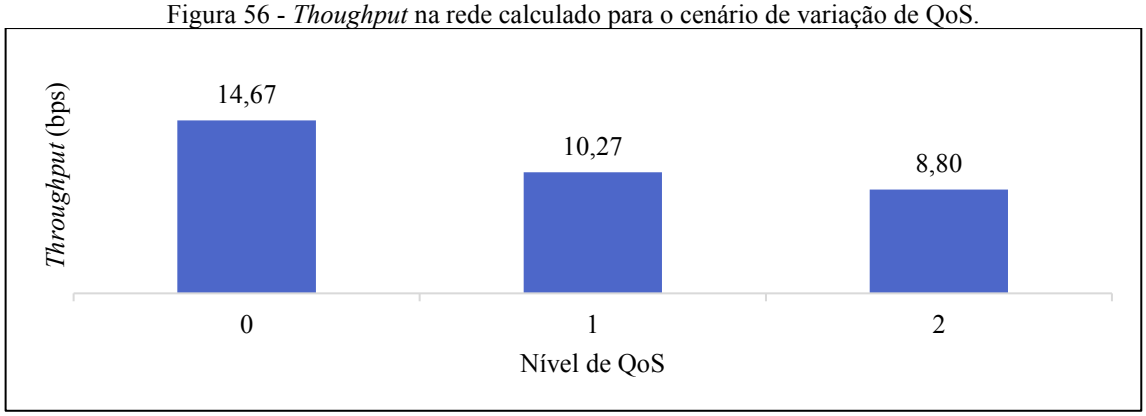


Fonte: Autoria Própria.

De acordo com a Figura 56, com o nível de QoS 0 tem-se o maior fluxo de dados, uma vez que não é necessário aguardar uma confirmação de entrega de mensagem. Nos níveis 1 e 2

de QoS, o *throughput* diminui, com base na definição dos níveis de controle e garantia de entrega de dados, que pode resultar em sobrecarga na rede e consequentemente, diminuição de *throughput*.

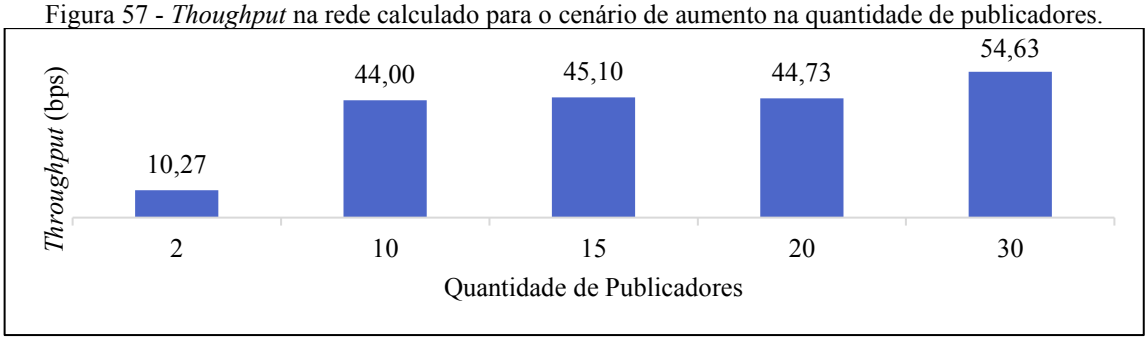
Figura 56 - *Thoughput* na rede calculado para o cenário de variação de QoS.



Fonte: Autoria Própria.

No cenário de aumento progressivo de publicadores (Figura 57), o *throughput* é diretamente proporcional à quantidade de publicadores na rede. Inicialmente, com apenas 2 publicadores, o *throughput* alcança seu menor valor do caso, mas aumenta de forma significante quando o número é elevado para 10 publicadores e posteriormente fica na faixa de 44 a 55 bps. Esse comportamento indica que a rede possui uma limitação, onde a adição de mais publicadores não resulta em ganho significativo devido ao aumento do número de mensagens simultâneas e a sobrecarga.

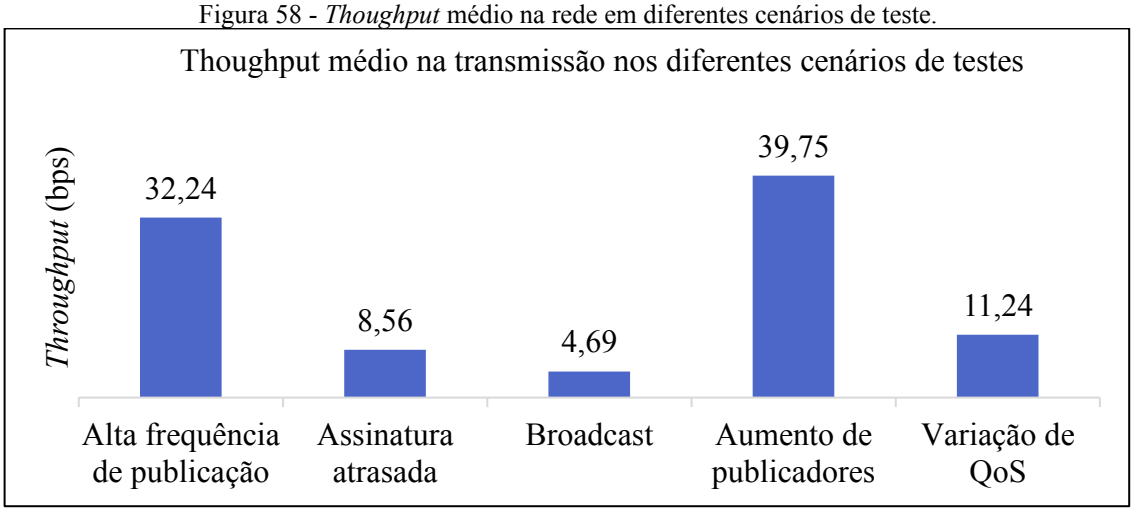
Figura 57 - *Thoughput* na rede calculado para o cenário de aumento na quantidade de publicadores.



Fonte: Autoria Própria.

Na Figura 58, é possível identificar que o cenário de variação de QoS tem o maior valor para o *throughput* médio, demonstrando como os níveis de qualidade de serviço influenciam diretamente no comportamento da rede. Os cenários de alta frequência de publicação e assinatura atrasada, em média, possuem valores intermediários, enquanto os cenários de *broadcast* e aumento de publicadores tem os menores valores.

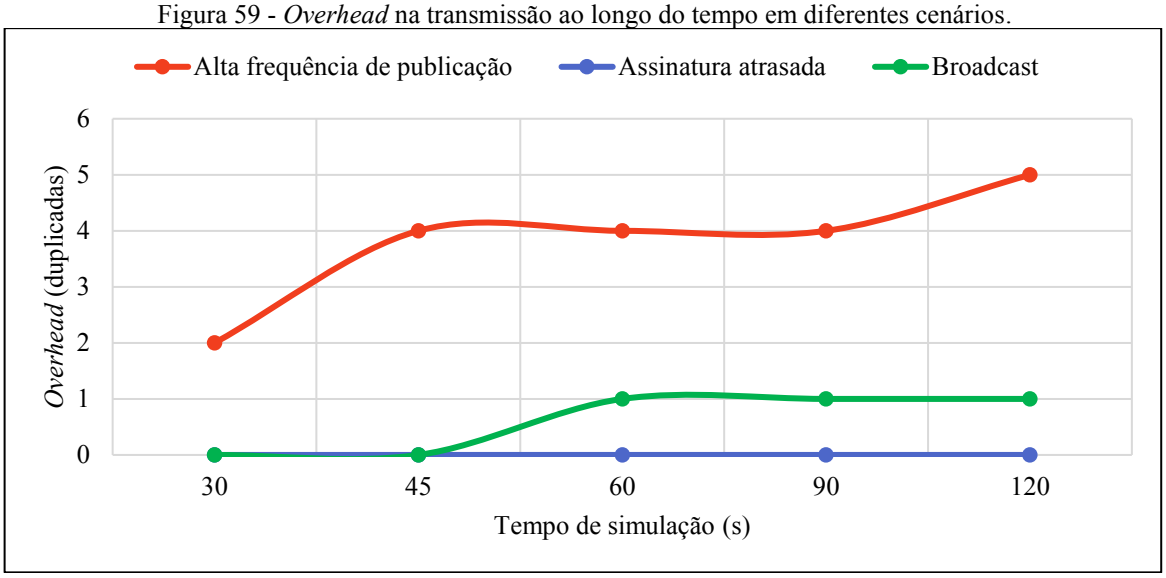
Figura 58 - *Thoughput* médio na rede em diferentes cenários de teste.



Fonte: Autoria Própria.

Acerca do *overhead* na transmissão, na Figura 59, observou-se que essa métrica tende a aumentar ao longo do tempo no cenário de alta frequência de publicação, sugerindo que esse cenário contribui para o aumento de tráfego na rede, resultando em maiores possibilidades de duplicadas e, consequentemente, no *overhead.* O cenário de assinatura atrasada apresentou praticamente zero *overhead* ao longo dos testes, devido à inscrição tardia, já que menos mensagens são enviadas aos assinantes no início, resultando em menor número de pacotes duplicados. O cenário de *broadcast* apresentou um pequeno aumento no *overhead* a partir dos 45 segundos, atingindo uma média de pouco mais de 1 duplicada. Esse aumento limitado no *overhead* está relacionado à natureza da transmissão em massa, que envia a mesma mensagem para muitos destinatários de forma eficiente.

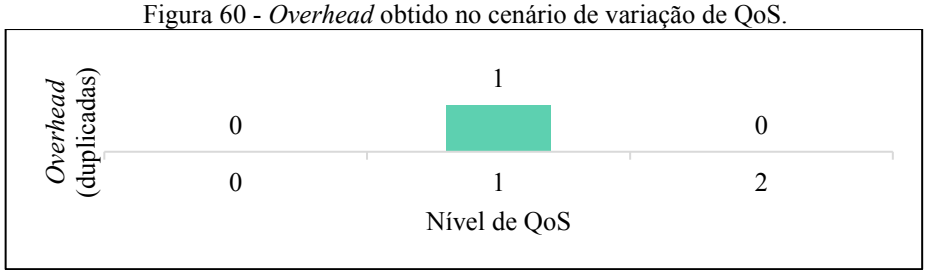
Figura 59 - *Overhead* na transmissão ao longo do tempo em diferentes cenários.



Fonte: Autoria Própria.

Na Figura 60, com os valores de *overhead* para o cenário de variação de QoS, é ilustrado que foi recebida apenas uma duplicada, com o nível de QoS 1, que gera confirmações na entrega de mensagens e às vezes, informações duplicadas.

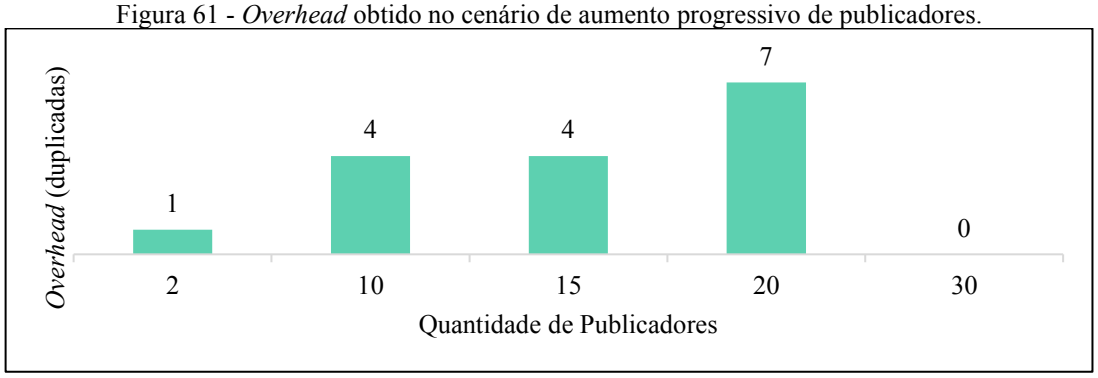
Figura 60 - *Overhead* obtido no cenário de variação de QoS.



Fonte: Autoria Própria.

Conforme a Figura 61, o número de publicadores aumenta de 2 para 20 e o *overhead* também aumenta significativamente, atingindo 7 duplicadas para 20 publicadores. Isso indica que um maior número de dispositivos publicando mensagens na rede contribui para a maior sobrecarga e duplicação de pacotes.

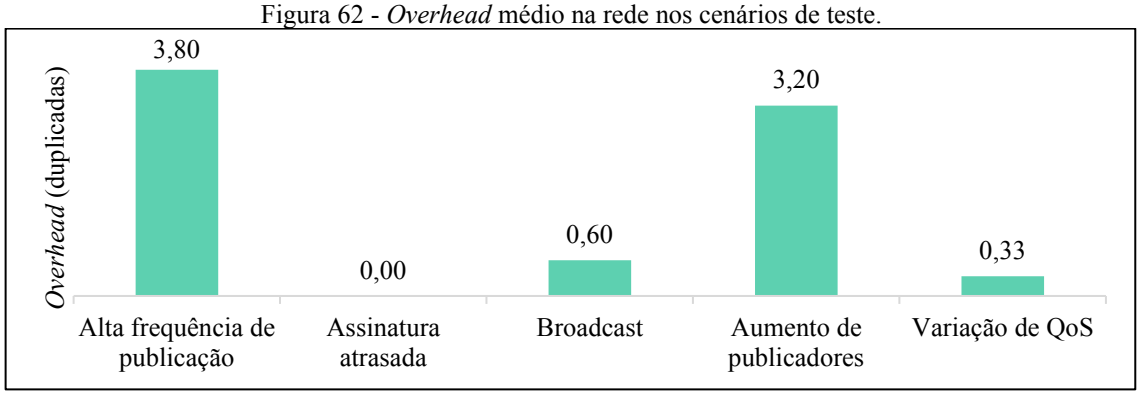
Figura 61 - *Overhead* obtido no cenário de aumento progressivo de publicadores.



Fonte: Autoria Própria.

De forma geral, conforme a Figura 62, nos cenários de assinatura atrasada, *broadcast* e variação de QoS não possuem *overhead* significativo, enquanto os cenários de alta frequência e de aumento de publicadores tem valores intermediários.

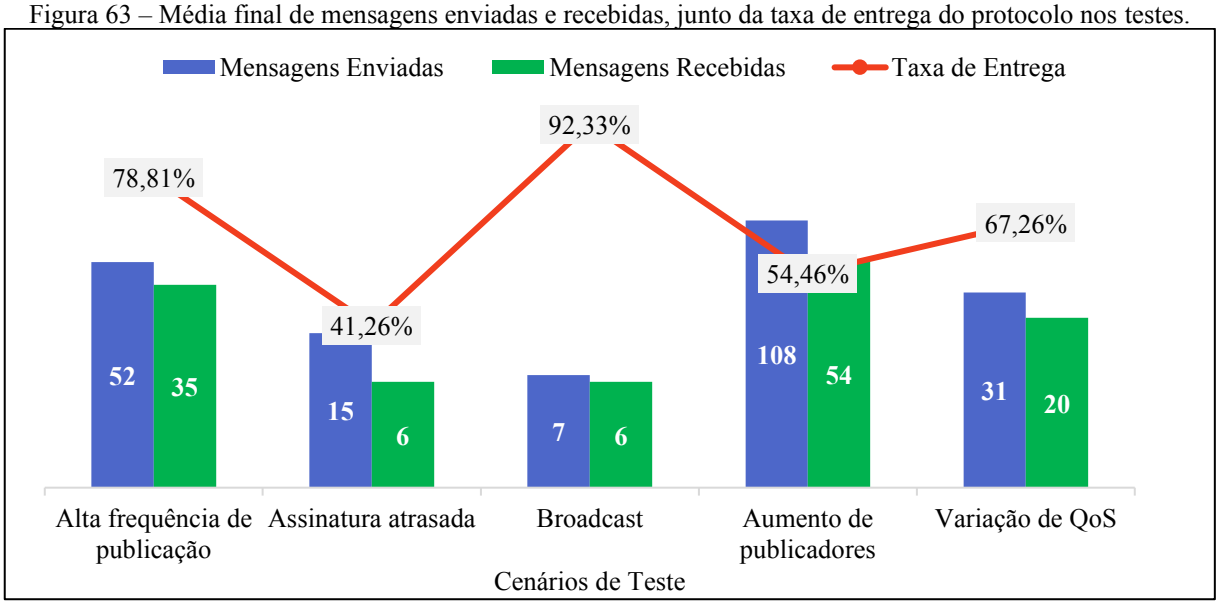
Figura 62 - *Overhead* médio na rede nos cenários de teste.



Fonte: Autoria Própria.

Na Figura 63 encontram-se três métricas principais: mensagens enviadas, mensagens recebidas e taxa de entrega de mensagens. Dessa maneira, o cenário de alta frequência de publicação apresenta uma taxa de entrega razoável, mas ainda houve uma perda de aproximadamente 21,19% das mensagens. Isso pode indicar que, devido ao grande número de mensagens sendo publicadas rapidamente, a rede ficou sobrecarregada e não conseguiu entregar todas as mensagens, resultando em perdas.

Figura 63 – Média final de mensagens enviadas e recebidas, junto da taxa de entrega do protocolo nos testes.



Fonte: Autoria Própria.

Ainda, nos cenários de assinatura atrasada e *broadcast* foram enviados uma menor quantidade de mensagens enviadas e recebidas, porém, no cenário de assinatura atrasada, tem-se a menor taxa de entrega de mensagens, com apenas 41,26% e no cenário de *broadcast*, tem-se a maior taxa de entrega, com quase 100% de eficiência na entrega de mensagens. Por fim, observou-se que o aumento na quantidade de publicadores na rede causou uma sobrecarga na rede, diminuindo a taxa de entrega de mensagens.

5 CONCLUSÕES

A presente pesquisa teve como objetivo analisar o desempenho do protocolo MQTT-SN em redes padronizadas pelo IEEE 802.11, no contexto de aplicações para a Internet das Coisas (IoT), utilizando o simulador OMNeT++. Para atingir esse objetivo, foram criados diversos cenários de testes que exploraram diferentes aspectos do comportamento do protocolo, incluindo frequência de publicação, atraso na assinatura, transmissão em *broadcast*, aumento do número de publicadores e variação na qualidade de serviço (QoS). A análise focou em métricas como latência, *throughput*, taxa de entrega de mensagens e *overhead*.

De acordo com a seção de resultados, foi possível identificar que o protocolo MQTT-SN tem seu desempenho diretamente influenciado pelas condições de rede e pela configuração dos parâmetros de comunicação. Cenários de alta frequência de publicação e aumento do número de publicadores resultaram em uma maior sobrecarga de rede e uma redução significativa na taxa de entrega de mensagens, evidenciando limitações da capacidade da rede em lidar com altos volumes de mensagens. Por outro lado, cenários de *broadcast* se mostraram eficazes em termos de entrega de mensagens, alcançando taxas de entrega superiores, o que reflete a vantagem da transmissão em massa em determinadas situações.

Do mesmo modo, a variação dos níveis de QoS revelou que, embora aumente a garantia de entrega das mensagens, também há um impacto na latência e no *throughput*, sendo necessário um balanço entre qualidade e desempenho, dependendo dos requisitos da aplicação IoT em questão. O cenário de assinatura atrasada mostrou uma significativa perda de mensagens, destacando a importância de uma sincronização adequada entre os publicadores e assinantes para otimizar a taxa de entrega.

Portanto, conclui-se que o protocolo MQTT-SN é uma opção viável para aplicações IoT, desde que haja uma configuração cuidadosa dos parâmetros de publicação e de QoS, e uma avaliação das condições de rede que garantam a melhor relação entre eficiência e desempenho. Em aplicações onde a latência é crítica ou onde o número de dispositivos é elevado, faz-se necessário explorar soluções que minimizem a sobrecarga da rede, tais como o ajuste dinâmico dos níveis de QoS e o uso de mecanismos de controle para mitigar o congestionamento.

REFERÊNCIAS BIBLIOGRÁFICAS

BASSI, Alessandro; EUROPE, Hitachi; HORN, Geir. Internet of Things in 2020: Roadmap for the future. **European Commission: Information Society and Media**, Bruxelas, n. 1, p. 4-6, Maio 2008.

COSMI, Arthur B.; MOTA, Vinícius F. S. **Uma Análise dos Protocolos de Comunicação para Internet das Coisas**. III Workshop de Computação Urbana. Porto Alegre: Sociedade Brasileira de Computação. 2019. p. 153-166.

GODOI, Maiko G. D.; ARAÚJO, Liriane S. D. A INTERNET DAS COISAS: evolução, impactos e benefícios. **Revista Interface Tecnológica**, São Paulo, v. 16, n. 1, p. 19-30, Junho 2019. ISSN 2447-0864.

GUMUS, Tayfun. **Implementation of MQTT-SN in OMNeT++**. Politecnico Milano 1863 - Scuola di Ingegnaria Industriale e Dell'informazione. Milão. 2024.

HERRERO, Rolando. MQTT-SN, CoAP, and RTP in wireless IoT real-time communications. **Multimedia Systems**, v. 26, p. 643–654, Julho 2020.

KHOROV, Evgeny; LEVITSKY, Ilya; AKYILDIZ, Ian F. Current Status and Directions of IEEE 802.11be, the Future Wi-Fi 7. **IEEE Access**, v. 8, p. 88664-88688, 2020.

KUROSE, James F.; ROSS, Keith W. **Redes de Computadores e a Internet:** Uma Abordagem Top-Down. 6ª. ed.

LEITE, J.R. E.; MARTINS, Paulo S.; URSINI, Edson L. **A Internet das Coisas (IoT):** Tecnologias e Aplicações. Brazilian Technology Symposium. [S.l.]: [s.n.]. 2017.

LOPEZ-AGUILERA, Elena; GARCIA-VILLEGAS, Eduard; CASADEMONT, Jordi. Evaluation of IEEE 802.11 coexistence in WLAN deployments. **Wireless Networks**, v. 25, p. 87-104, 17 Junho 2019.

MACEDO, Ricardo T. et al. **Redes de Computadores**.

MAGRANI, Eduardo. **A Internet das Coisas**. 1. ed.

MEDEIROS, Flaviani S. B. et al. Internet das Coisas: uma investigação do conhecimento científico em artigos acadêmicos na última década. **Revista Eletrônica de Administração e Turismo**, v. 12, n. 7, p. 1652-1674, Dezembro 2018.

MINERVA, Roberto; BIRU, Abyi; ROTONDI, Domenico. **Towards a definition of the Internet of Things (IoT)**. 1ª. ed.

MISHRA, Biswajeeban; KERTESZ, Attila. The Use of MQTT in M2M and IoT Systems: A Survey. **IEEE Access**, v. 8, p. 201071-201086, 2020.

MOON, Bernard. **Internet of Things & Hardware Industry Report**. SparkLabs Global Ventures. Califórnia, p. 3-4. 2016.

OPENSIM LTD. A Quick Overview of the OMNeT++ IDE. **OMNeT++**, 2019. Disponível em: <https://omnetpp.org/documentation/ide-overview/>. Acesso em: 1 Dezembro 2024.

OPENSIM LTD. What is OMNeT++? **OMNeT++**, 2019. Disponível em: <https://omnetpp.org/intro/>. Acesso em: 1 Dezembro 2024.

PEREIRA, Felisberto et al. Challenges in Resource-Constrained IoT Devices: Energy and Communication as Critical Success Factors for Future IoT Deployment. **Sensors**, v. 20, n. 6420, 10 Novembro 2020.

QUINCOZES, S.; TUBINO, E.; KAZIENKO, J. MQTT Protocol: Fundamentals, Tools and Future. **IEEE Latin America Transactions**, v. 17, n. 9, p. 1439-1448, September 2019.

SANTOS, Bruno P. et al. Internet das Coisas - da Teoria à Prática. **XXXIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos**, Salvador, Junho 2016.

SEVERINO, Antônio J. **Metodologia do Trabalho Científico**. 1ª. ed.

SINGH, Manisha; BARANWAL, Gaurav. **Quality of Service (QoS) in Internet of Things**. 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU). Bhimtal: [s.n.]. 2018. p. 1-6.

SONI, Dipa; MAKWANA, Ashwin. **A SURVEY ON MQTT:** A PROTOCOL OF INTERNET OF THINGS(IOT). International conference on telecommunication, power analysis and computing techniques (ICTPACT). [S.l.]: [s.n.]. 2017. p. 173-177.

SPOHN, Marco A.; GENERO, Willian B. Análise experimental dos protocolos MQTT e MQTT-SN. **Revista Brasileira de Computação Aplicada**, Passo Fundo, v. 15, n. 1, p. 22-33, Abril 2023.

TANENBAUM, Andrew S.; WHETERALL, David. **Redes de Computadores**. 5. ed.

TOLDINAS, Jevgenijus et al. MQTT Quality of Service versus Energy Consumption. **2019 23rd International Conference Electronics**, p. 1-4, 2019.

YASSEIN, Muneer B. et al. Internet of Things: Survey and open issues of MQTT protocol. **2017 International Conference on Engineering & MIS (ICEMIS)**, Monastir, p. 1-6, 2017.

ZAHOOR, Saniya; MIR, Roohie N. Resource management in pervasive Internet of Things: A survey. **Journal of King Saud University – Computer and Information Sciences**, v. 33, n. 8, p. 921-935, Agosto 2018.