Análise de Desempenho e Planejamento de Capacidade em Arquiteturas MQTT para Aplicações IoT

Melissa Alves [1], José Wanderlei[1], Luiz Nelson Lima[1], Arthur Sabino[1], Vandirleya Barbosa[1], Luis Guilherme Silva[1], Leonel Feitosa[1], Rodrigo Righi[2], Francisco Airton Silva[1]

[1]Laboratório de Pesquisa Aplicada a Sistemas Distribuídos (PASID), Universidade Federal do Piauí (UFPI), Picos, Piauí

[2]Unisinos, Porto Alegre, Rio Grande do Sul

{melissaalves,jose.rocha,luizznelson,arthursabino}@ufpi.edu.br

{vandirleya.barbosa,luis.e,leonelfeitosa,faps}@ufpi.edu.br

rrrighi@unisinos.br

Resumo. *O protocolo MQTT (Message Queuing Telemetry Transport) é amplamente adotado em aplicações de Internet das Coisas (IoT) devido à sua leveza, que o torna adequado para dispositivos com recursos limitados. No entanto, em ambientes complexos, surgem desafios relacionados à configuração de brokers e ao gerenciamento do tráfego de mensagens, que podem comprometer o desempenho das arquiteturas. Este artigo apresenta um modelo baseado em Redes de Petri Estocásticas (SPN) para análise e planejamento de capacidade em arquiteturas MQTT, permitindo avaliar o comportamento do sistema sob diferentes níveis de tráfego e configurações de brokers. Os resultados indicam que o tempo de resposta e a vazão são diretamente afetados pela intensidade do tráfego. Essa análise fornece uma base sólida para decisões estratégicas de configuração e expansão de sistemas MQTT em cenários críticos.*

Abstract. *The MQTT (Message Queuing Telemetry Transport) protocol is widely adopted in Internet of Things (IoT) applications due to its lightness, which makes it suitable for devices with limited resources. However, in complex environments, challenges arise related to the configuration of brokers and the management of message traffic, which can compromise the performance of architectures. This article presents a model based on Stochastic Petri Nets (SPN) for capacity analysis and planning in MQTT architectures, allowing the system's behavior to be evaluated under different traffic levels and broker configurations. The results indicate that response time and throughput are directly affected by traffic intensity. This analysis provides a solid basis for strategic decisions on the configuration and expansion of MQTT systems in critical scenarios.*

1. Introdução

A Internet das Coisas (IoT) representa um paradigma transformador, possibilitando a conectividade entre bilhões de dispositivos inteligentes, sensores e sistemas em diversos setores. Essa conectividade revoluciona processos, otimiza recursos e melhora a qualidade de vida por meio da troca de dados em tempo real e da automação inteligente

[Raja et al. 2023]. Estima-se que o número de dispositivos IoT ultrapassará 29 bilhões até 2030, consolidando sua presença em diferentes áreas da sociedade [Vailshery 2024]. O tráfego gerado por esses dispositivos, impulsionado por tecnologias como 5G e 6G, é fundamental para setores como saúde, transporte e energia, promovendo maior eficiência e interação com o ambiente [Laghari et al. 2024]. Dentre os protocolos de comunicação para IoT, o MQTT se destaca por sua leveza, escalabilidade e eficiência em ambientes com conectividade intermitente [Doshi et al. 2024]. Seu design prioriza o uso reduzido de largura de banda e baixa exigência de recursos dos dispositivos, ao mesmo tempo em que assegura a entrega confiável de mensagens [Mishra and Kertesz 2020].

A crescente complexidade dos ambientes IoT impõe desafios ao desempenho da comunicação. Parâmetros como níveis de qualidade de serviço (QoS), tamanhos de pacotes e intervalos de manutenção da conexão influenciam diretamente a escalabilidade e confiabilidade do sistema [Spohn 2022, Dizdarević et al. 2019]. Nesse contexto, torna-se fundamental compreender como essas configurações impactam o comportamento do protocolo MQTT sob diferentes condições operacionais, especialmente em termos de utilização de recursos e qualidade de serviço.

Embora diversos estudos tenham investigado o desempenho do MQTT, muitos se restringem a ambientes controlados ou simulações teóricas, focando em métricas como vazão e latência [Mishra et al. 2021, Nast et al. 2023]. Testes de estresse foram utilizados para avaliar o comportamento sob diferentes cargas [Rodriguez and Batista 2023], enquanto outras abordagens exploraram arquiteturas híbridas com Kafka [Nam et al. 2022, Yokotani et al. 2021] ou a resiliência de brokers distribuídos [Gruener et al. 2021]. No entanto, poucos trabalhos abordam a influência da variabilidade de tráfego em tempo de execução, considerando cenários de carga variável e os impactos diretos sobre o desempenho da arquitetura MQTT.

Este artigo propõe um modelo baseado em Redes de Petri Estocásticas (SPN) para representar e analisar o funcionamento do protocolo MQTT em arquiteturas IoT. As SPNs permitem capturar características dinâmicas, concorrência, atrasos e limitações de capacidade, oferecendo uma estrutura robusta para modelar e avaliar sistemas estocásticos. Essa abordagem é amplamente aplicada em domínios como redes de computadores, sistemas de manufatura e IoT [Maciel 2023]. Assim, as principais contribuições deste trabalho incluem: (i) a modelagem SPN de uma arquitetura MQTT, incorporando elementos como fila de mensagens, limitação de capacidade e atrasos temporais; (ii) a análise de métricas fundamentais, como tempo médio de resposta (MRT), vazão (TP), utilização de recursos e probabilidade de descarte de mensagens (DP), com foco em configurações realistas de brokers; e (iii) a identificação de gargalos e limites operacionais sob diferentes intensidades de tráfego, contribuindo para o planejamento de capacidade e a otimização de infraestrutura em sistemas baseados em MQTT.

O restante deste trabalho está organizado da seguinte forma: A Seção 2 apresenta trabalhos relacionados por meio de uma tabela comparativa entre os estudos considerados. A Seção 3 apresenta a arquitetura considerada neste trabalho. A Seção 4 apresenta o modelo SPN proposto e as métricas utilizadas para a avaliação de desempenho. Na Seção 5, os resultados de dois estudos de caso são apresentados. A Seção 6 apresenta as conclusões e trabalhos futuros.

2. Trabalhos Relacionados

Esta seção apresenta treze trabalhos relacionados. A Tabela 1 resume as contribuições dos trabalhos e os critérios de comparação. Os seguintes critérios de comparação foram adotados: contexto, método de avaliação, métricas e variações de configuração.

Tabela 1. Trabalhos relacionados.

Trabalho	Contexto	Métricas	Variação nas Configurações de Comunicação	Método de Avaliação
[Hafaiedh 2022]	Análise de protocolos	Número de mensagens recebidas e enviadas, Tempo de verificação das propriedades CTL	✓	Simulação
[Nast et al. 2023]	Comunicação M2M	Vazão das mensagens, Atraso de publicação, Latência de ponta a ponta	✓	Experimentação
[Putpuek et al. 2023]	Smart Agriculture	Tempo entre endpoints, Tempo de transmissão, Desvio padrão, Tempo mínimo e máximo para OPC UA	✓	Experimentação
[Seoane et al. 2021]	Segurança	Largura de banda, Utilização, Latência, Consumo de energia	✓	Experimentação
[Yokotani et al. 2021]	Conceito	Atraso Médio	✓	Modelo de filas
[Gemirter et al. 2021]	Análise de protocolos	Latência de mensagens, Utilização, Largura de banda, Consumo de energia	×	Simulação
[Rodriguez and Batista 2023]	Smart Cities	Utilização, Tráfego de rede e troca de pacotes	✓	Experimentação
[Gruener et al. 2021]	Comunicação Distribuída	Perda de mensagens, latência, duplicação e mensagens fora de ordem	✓	Simulação
[Li and Fujita 2024]	Smart transportation	Latência, Utilização	✓	Simulação
[Hmissi and Ouni 2022]	Comunicação Distribuída	Latência, Sobrecarga de assinatura de dados	×	Simulação
[Mishra et al. 2021]	Comunicação M2M	Utilização, Latência, Taxa de mensagens	✓	Experimentação
[Zorkany et al. 2019]	E-Health	Tempo de envio das mensagens, Média de bytes usados, Latência	✓	Simulação
[Nam et al. 2022]	Cloud computing	Tempo de processamento, Tempo de transmissão e TP	✓	Experimentação
Este Trabalho	Conceito	MRT, TP, DP e utilização	✓	Modelo SPN

Os trabalhos categorizados em Simulação avaliam o desempenho de protocolos IoT em cenários controlados. Por exemplo, [Hafaiedh 2022] usa o simulador UPPAAL para verificar propriedades QoS de MQTT, AMQP e CoAP, enquanto [Hmissi and Ouni 2022] propõe o sistema TD-MQTT, que distribui brokers para melhorar a escalabilidade em ambientes IoT. Nos estudos de Experimentação, o foco está sobre a validação prática de arquiteturas e protocolos. Por exemplo, [Nast et al. 2023] avalia brokers MQTT-SN em plataformas Raspberry Pi, e [Mishra et al. 2021] realiza testes de estresse em seis brokers MQTT para identificar limitações sob alta carga. Na categoria

de Modelagem, [Yokotani et al. 2021] destaca-se com sua arquitetura de anel virtual para brokers MQTT, utilizando modelos de filas para otimizar atrasos de transferência.

O nosso trabalho se destaca pela análise abrangente de métricas, incluindo MRT, TP e DP, superando a limitação de estudos anteriores que se concentram em um conjunto restrito de indicadores. Além disso, avaliamos diferentes configurações de envio de mensagens e seu impacto sobre a camada de subscribers, permitindo identificar gargalos e comportamentos críticos em cenários de carga variável. Essa abordagem contribui para preencher lacunas na literatura e oferece uma base sólida para a modelagem e otimização do protocolo MQTT em arquiteturas mais complexas e realistas.

3. Arquitetura

Esta seção descreve a arquitetura utilizada para análise do protocolo MQTT. A Figura 1 apresenta os principais componentes e o fluxo de funcionamento da comunicação. Três configurações distintas foram consideradas, variando a quantidade de tópicos e subscribers conectados a um mesmo publisher. A comunicação é estruturada em três camadas principais: publishers, broker e subscribers, com uma camada adicional dedicada ao armazenamento final dos dados processados.

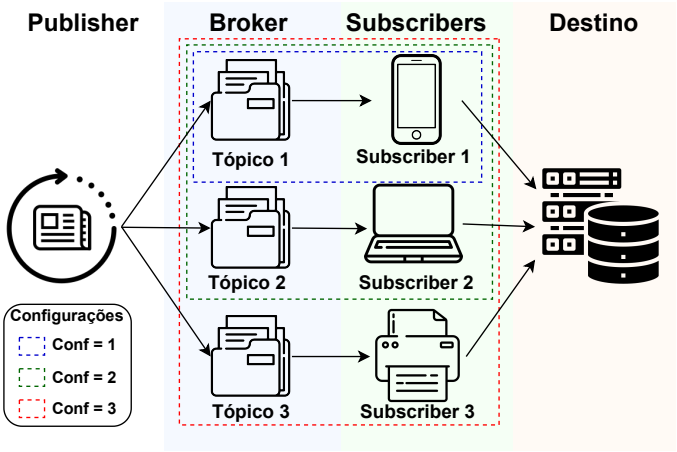


Figura 1. Arquitetura do protocolo MQTT aplicada ao modelo SPN.

Na primeira camada, os publishers são responsáveis por gerar e publicar dados em tópicos específicos, sem qualquer comunicação direta com os subscribers. Essa separação lógica promove o desacoplamento entre produção e consumo de mensagens, característica fundamental do modelo pub/sub. O broker MQTT atua como o núcleo da arquitetura, sendo responsável por gerenciar, encaminhar e armazenar as mensagens recebidas. Ele também aplica políticas de Qualidade de Serviço (QoS) que garantem entrega confiável mesmo em condições adversas de rede [Fiege et al. 2006]. O broker armazena mensagens temporariamente, garantindo que novos subscribers recebam dados históricos se necessário. Para isso, requer capacidade de armazenamento adequada e gerenciamento eficiente de recursos, evitando perdas e congestionamentos.

A terceira camada é composta pelos subscribers, que se inscrevem nos tópicos de interesse. Cada subscriber processa mensagens conforme seu papel na aplicação, podendo lidar com diferentes tópicos ou informações. A comunicação entre publisher e subscriber

é intermediada exclusivamente pelo broker, de forma assíncrona, utilizando identificadores únicos de tópico. A arquitetura suporta três configurações operacionais, permitindo: (1) publicação em um único tópico com um único subscriber; (2) um publisher publicando em múltiplos tópicos; (3) distribuição entre múltiplos subscribers conectados a diferentes tópicos. Essa flexibilidade permite adaptar o sistema a diferentes volumes de tráfego e necessidades de processamento, otimizando o desempenho conforme a complexidade da aplicação. Após o processamento pelos subscribers, os dados são transferidos de forma segura para um banco de dados, compondo a camada final de armazenamento, voltada à persistência e análise futura.

Para fins de modelagem, algumas premissas simplificadoras foram adotadas: distribuição uniforme de dados entre os tópicos, capacidade homogênea de tópicos e subscribers, rede com disponibilidade constante e ausência de perdas ou atrasos. Apesar disso, o modelo pode ser facilmente estendido para representar ambientes mais realistas, incluindo falhas, latências variáveis e comportamento dinâmico da rede, ampliando a aplicabilidade da arquitetura a contextos IoT mais exigentes.

4. Modelo SPN

Esta seção apresenta a estrutura do modelo SPN, com detalhes sobre o fluxo de execução. A descrição segue a sequência de camadas que o modelo apresenta. Para a construção do modelo e realização das simulações, utilizou-se a ferramenta Mercury [Maciel et al. 2017].

4.1. Estrutura do Modelo

A Figura 2 apresenta o modelo SPN baseado na arquitetura anteriormente discutida. O modelo visa auxiliar no planejamento de capacidade de arquiteturas MQTT utilizadas com IoT. Para isso, propõe-se um modelo SPN para simular diferentes cenários de tráfego de dados e auxiliar a identificar gargalos ou regiões mais sensíveis a impactar no desempenho do sistema.

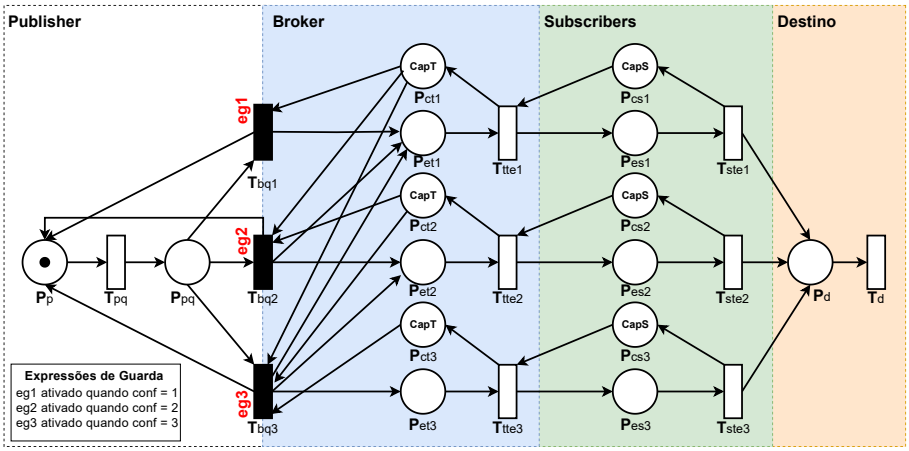


Figura 2. Modelo SPN que modela o comportamento do protocolo MQTT.

O modelo proposto possui quatro camadas: publisher, broker, subscribers e destino. A camada publisher gera os dados que alimentam o sistema, enquanto o broker gerencia a comunicação, organizando as informações em tópicos acessados de forma

assíncrona pelos subscribers. A camada subscriber utiliza o protocolo MQTT para receber, analisar e encaminhar dados à camada destino, responsável pelo armazenamento ou uso final.

Para simplificar, algumas abreviações foram feitas: P (lugares), T (transições), Cap (lugares marcados), e funções específicas em minúsculo, como p (publisher), q (fila), b (broker), t (tópico), c (capacidade), e (envio), te (tempo de envio) e d (destino). Expressões de guarda (Eg) são condições booleanas que controlam transições, ativadas apenas se verdadeiras [Maciel 2020]. O modelo tem três configurações definidas por `eg1`, `eg2` e `eg3`, vinculadas às transições `Tbq1`, `Tbq2` e `Tbq3`. A variável $conf$ ativa as configurações: $conf = 1$ habilita `Tbq1`, $conf = 2$ habilita `Tbq1` e `Tbq2`, e $conf = 3$ habilita as três transições.

A Tabela 2 descreve os elementos do modelo. A operação começa em Pp, representando o publisher aguardando dados gerados pelos dispositivos. Esses dados entram em uma fila para serem enviados ao broker, processo representado por `Tpq`, que modela os intervalos entre as chegadas. A partir dessa estrutura, o modelo organiza a transmissão, processamento e destino dos dados de maneira flexível e configurável.

Tabela 2. Descrição dos componentes do modelo.

Tipo	Componente(s)	Descrição
Lugar	**Pp**	Entrada de dados pelo dispositivo do publisher.
	Ppq	Disponibilidade de fila para os dados do publisher.
	Pct1, Pct2, Pct3	Capacidade dos tópicos no broker.
	Pcs1, Pcs2, Pcs3	Capacidade de armazenamento das mensagens nos subscribers.
	Pet1, Pet2, Pet3	Envio das mensagens dos tópicos para os subscribers correspondentes.
	Pes1, Pes2, Pes3	Envio das mensagens dos subscribers para o destino.
	Pd	Destino dos dados dos subscribers.
Transição Temporizada	**Tpq**	Tempo entre as chegadas dos dados no publisher.
	Ttte1, Ttte2, Ttte3	Tempo de envio das mensagens dos tópicos para os subscribers.
	Tste1, Tste2, Tste3	Tempo de envio das mensagens dos subscribers para o destino.
	Td	Tempo para os dados serem armazenados em um banco de dados.
Transição Imediata	**Tbq1, Tbq2, Tbq3**	Encaminham as mensagens do publisher para o broker, conforme a configuração estabelecida.
Lugar Marcado	**CapT e CapS**	Capacidade máxima de fila dos tópicos e dos subscribers.

No broker, os dados são distribuídos entre três tópicos: *Pct1*, *Pct2* e *Pct3*, que indicam a capacidade dos tópicos. Os componentes *Pet1*, *Pet2* e *Pet3* representam o envio das mensagens desses tópicos para os subscribers correspondentes. A capacidade de envio (*Pet*) é limitada pela capacidade dos tópicos (*Pct*) e pelo lugar marcado *CapT*. A distribuição das mensagens é definida pela configuração do broker (*conf*). Na configuração 1, a expressão *eg1* habilita *Tbq1*, enviando mensagens de *Pct1* para o subscriber *Pcs1* em um único caminho. Na configuração 2, *eg2* habilita *Tbq1* e *Tbq2*, permitindo dois caminhos: mensagens de *Pct1* para *Pcs1* e de *Pct2* para *Pcs2*. Já na configuração 3, *eg3* habilita *Tbq1*, *Tbq2* e *Tbq3*, criando três caminhos: mensagens de *Pct1* para *Pcs1*, *Pct2* para *Pcs2* e *Pct3* para *Pcs3*.

Após serem distribuídas nos tópicos, as mensagens são encaminhadas para os subscribers. Este processo é modelado pelas transições temporizadas *Ttte1*, *Ttte2*, *Ttte3*, que representam o tempo necessário para enviar as mensagens dos tópicos no broker para os lugares correspondentes aos subscribers. Os subscribers possuem lugares de capacidade de armazenamento, representados por *Pcs1*, *Pcs2* e *Pcs3*, onde as mensagens são

armazenadas temporariamente. As transições temporizadas *Tste1*, *Tste2* e *Tste3* modelam o envio das mensagens desses lugares de armazenamento para o destino, movendo *tokens* para *Pd*, onde as mensagens dos subscribers são armazenadas em um banco de dados.

4.2. Métricas

As métricas utilizadas para avaliar o desempenho do sistema incluem: utilização de recursos da camada de broker, tempo médio de resposta (MRT), probabilidade de descarte de dados e vazão do sistema. A obtenção dessas métricas foi realizada por meio do modelo SPN.

A utilização, calculada pela Equação (1), avalia o uso dos recursos, identificando gargalos ou ociosidade. No modelo, foca-se na camada de subscribers, calculando a média de *tokens* nos subscribers dividida pela sua capacidade, expressa em porcentagem.

$$\text{Utilização} = \left(\frac{Esp(Pcs1) + Esp(Pcs2) + Esp(Pcs3)}{CapS \times 3} \right) \times 100 \tag{1}$$

A Equação (2) calcula a métrica DP, que indica a probabilidade de descarte de dados no sistema. Essa métrica avalia se todos os tópicos do broker estão cheios e ainda há dados prontos para envio na fila do publisher. O valor resultante, expresso em porcentagem, reflete a ineficiência no tratamento de grandes volumes de dados, apontando falhas na entrega por indisponibilidade de recursos.

$$\text{DP} = P\left\{(Pct1 = 0)AND(Pct2 = 0)AND(Pct3 = 0)AND(Ppq > 0)\right\} \times 100 \tag{2}$$

A Equação (3) calcula o TP, que representa a taxa média de dados entregues ao destino final dos subscribers. No modelo, é obtido dividindo a esperança de *tokens* em *Pd* pelo tempo médio de serviço *TS*, associado ao armazenamento dos dados em *Td*.

$$TP = \frac{Esp(\text{Pd})}{\text{TS}} \tag{3}$$

A Equação (4), baseada na lei de Little [Little and Graves 2008], calcula o MRT como o tempo médio necessário para que cada dado gerado seja enviado aos subscribers e processado pelo sistema. Esse cálculo, amplamente utilizado na teoria de filas, soma as esperanças dos *tokens* nas filas e divide pelo valor da vazão do sistema, avaliando a eficiência em relação ao tempo de resposta [Silva et al. 2021].

$$\text{MRT} = \frac{\sum_{i=1}^{3} \left(\mathbb{E}[Pet_i] + \mathbb{E}[Pes_i]\right)}{TP} \tag{4}$$

5. Estudos de Caso

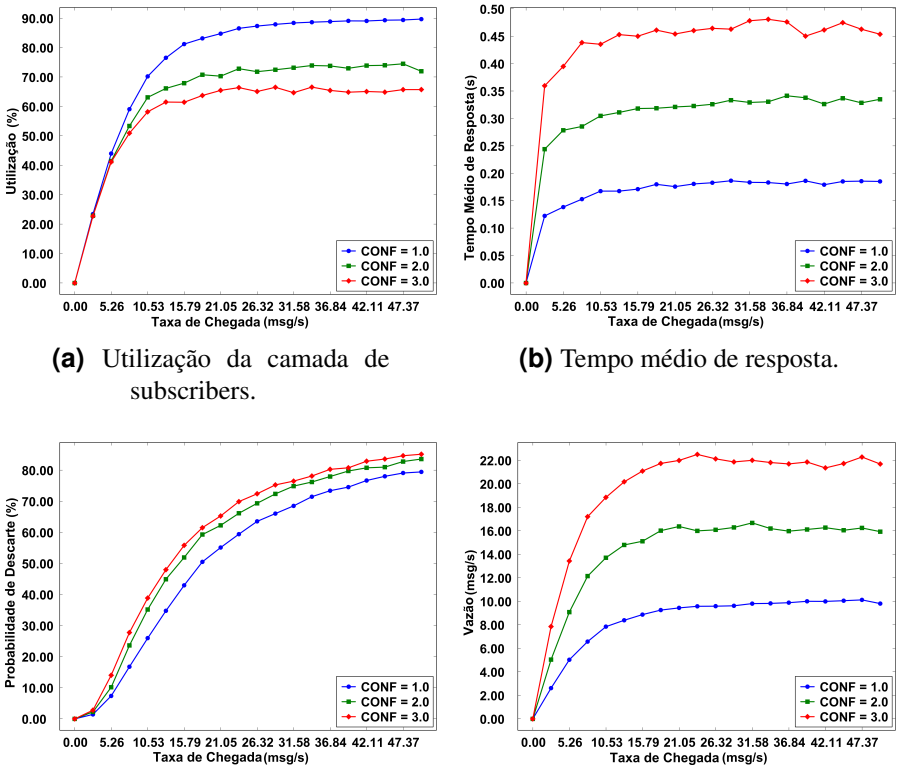
Esta seção apresenta dois estudos de caso desenvolvidos para avaliar o tráfego de dados no protocolo MQTT. O primeiro cenário explora diferentes configurações do modelo, enquanto o segundo investiga a influência combinada de dois fatores-chave no desempenho

do sistema: a capacidade dos subscribers (CapS) e o tempo de salvamento (STS). Ambos os estudos buscam analisar como variações nos recursos das camadas afetam a qualidade de serviço (QoS) e a eficiência da comunicação.

Os parâmetros do modelo incluem transições temporizadas representando os principais tempos de processamento: STT (0.50 s), STS (0.01 s) e TS (0.09 s). O parâmetro STT refere-se ao tempo necessário para que os dados sejam transmitidos do tópico no broker até os subscribers. O tempo STS corresponde ao intervalo até que os dados cheguem ao banco de dados, enquanto TS representa o tempo de escrita efetiva dos dados na base persistente. Os lugares CapT e CapS, que representam as capacidades de processamento dos tópicos e dos subscribers, respectivamente, são inicialmente configurados com valor igual a 1, simulando um cenário com recursos limitados.

5.1. Caso 1 - Variação na Configuração

No primeiro cenário, os resultados foram avaliados considerando uma variação na configuração utilizada. Os resultados obtidos são apresentados na Figura 3. As configurações usadas foram 1, 2 e 3 (Seção 4).



(a) Utilização da camada de subscribers.

(b) Tempo médio de resposta.

(c) Probabilidade de descarte.

(d) Vazão de dados.

Figura 3. Variação das configurações.

A configuração 1 apresentou a maior utilização da camada de subscribers, atingindo aproximadamente 90% para taxas de chegada acima de 10.53 msgs/s. O MRT permaneceu baixo, abaixo de 0.20 s, indicando alta eficiência no processamento sob cargas leves. No entanto, a probabilidade de descarte aumentou rapidamente a partir de 5.26 msgs/s, chegando a cerca de 80%, o que limitou a vazão a 8.00 msgs/s. Esses resultados

sugerem que essa configuração é adequada para cargas leves, mas não sustenta desempenho em cenários de alta demanda.

Na configuração 2, observou-se um comportamento intermediário. A utilização da camada de subscribers estabilizou-se em torno de 70% para 10.53 msgs/s, demonstrando melhor adaptação ao aumento da carga. O MRT aumentou para 0.30 s, e o DP cresceu de forma gradual, permanecendo abaixo de 80% até 36.84 msgs/s. O TP atingiu um pico de 16.00 msgs/s, o dobro da configuração anterior, indicando uma maior capacidade de processamento sob volumes moderados de tráfego.

A configuração 3 apresentou a menor utilização dos subscribers, estabilizando-se em 60%, devido à maior distribuição de mensagens entre três tópicos distintos. Como consequência, o MRT foi o mais elevado entre as configurações, alcançando aproximadamente 0.45 s. Apesar disso, essa configuração obteve o maior TP, chegando a 22.00 msgs/s, e uma DP superior à da configuração 2, evidenciando maior robustez em cenários com tráfego intenso, mesmo com aumento na latência.

De forma geral, os resultados mostram que existe uma relação de compensação entre a eficiência no processamento das mensagens (medida pelo tempo médio de resposta – MRT) e a capacidade do sistema de lidar com cargas elevadas (avaliada pela vazão – TP – e pela taxa de descarte – DP). A configuração 1 é mais indicada para cenários com baixo volume de mensagens e exigência de baixa latência. A configuração 2 oferece um equilíbrio entre desempenho e escalabilidade, sendo adequada para cargas moderadas. Já a configuração 3, embora apresente um tempo de resposta maior, é a mais recomendada para aplicações com alto volume de tráfego, onde robustez e capacidade de processamento são mais importantes do que a latência. A escolha da configuração mais adequada dependerá dos requisitos específicos de cada aplicação, especialmente no que diz respeito ao desempenho esperado e aos custos operacionais envolvidos.

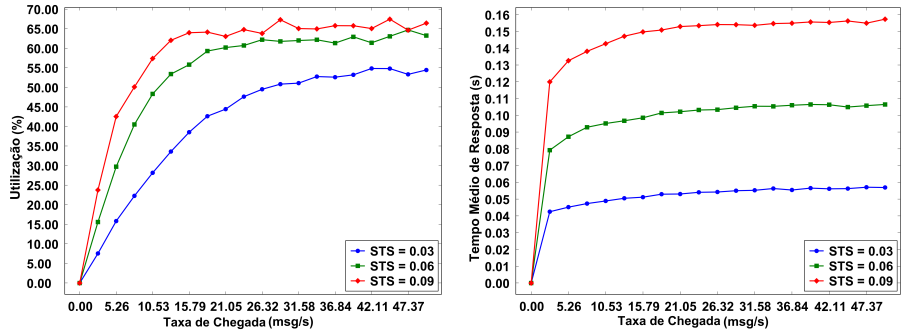
5.2. Caso 2 - Variação no Tempo de Serviço dos Subscribers

No segundo cenário, os resultados foram avaliados considerando a variação no tempo de serviço dos subscribers. Os resultados obtidos são apresentados na Figura 4. A variação no tempo de serviço dos subscribers foi de 0.03 s, 0.06 s, 0.09 s.

Com tempo de serviço de 0.03 s, a utilização da camada de subscribers foi a menor entre as configurações, cerca de 55% para uma taxa de chegada de 47.37 msgs/s. O MRT também foi o mais baixo, em torno de 0.04 s, indicando alta eficiência no processamento. Apesar do aumento gradual da probabilidade de descarte em taxas mais altas, essa configuração atingiu o maior TP, de 55.00 msgs/s, demonstrando excelente desempenho em termos de capacidade de processamento, mesmo sob carga elevada.

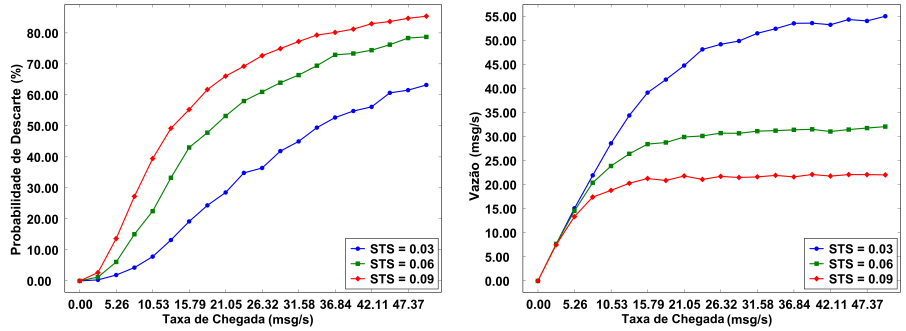
Com tempo de serviço de 0.06 s, a utilização dos subscribers aumentou para 70%. O MRT subiu para 0.09 s, e o DP alcançou 80% nas maiores taxas de chegada, o que limitou o TP a 30.00 msgs/s. Essa configuração apresentou bom desempenho em cargas médias, com processamento relativamente estável, mas mostrou limitações para lidar com cenários de alta demanda.

Na configuração com tempo de serviço de 0.09 s, a utilização dos subscribers foi a mais alta, 65%, mas o MRT também foi o maior, chegando a 0.16 s. O DP superou 80%, e o TP foi o menor entre as configurações, atingindo apenas 20.00 msgs/s. Esses resultados



(a) Utilização da camada de subscribers.

(b) Tempo médio de resposta.



(c) Probabilidade de descarte.

(d) Vazão de dados.

Figura 4. Variação do tempo de serviço dos subscribers.

indicam que, embora mais recursos sejam utilizados, a latência aumentada e o elevado descarte de mensagens tornam essa configuração inadequada para ambientes com tráfego intenso.

Os resultados mostram que o tempo de serviço impacta diretamente a eficiência e o uso de recursos do sistema. A configuração com 0.03 s proporcionou o melhor desempenho em termos de throughput e tempo de resposta, mas exigiu maior capacidade para conter o crescimento do DP. A configuração com 0.06 s apresentou um equilíbrio aceitável em cenários de carga moderada. Já o tempo de serviço de 0.09 s, embora com maior uso de recursos, comprometeu o desempenho global. Portanto, os resultados indicam que o tempo de serviço influencia diretamente o desempenho, sendo importante que a aplicação esteja preparada para lidar com os impactos causados por diferentes níveis de carga.

6. Conclusão

Este trabalho apresentou um modelo baseado em Redes de Petri Estocásticas para avaliar o desempenho do protocolo MQTT em sistemas IoT, considerando diferentes configurações e métricas como tempo médio de resposta, vazão, probabilidade de descarte e utilização de recursos. No primeiro cenário, configurações mais robustas mostraram-se adequadas para aplicações com alta demanda de processamento, embora com maior latência e uso de recursos. Configurações mais simples, por sua vez, priorizam baixa latência e eficiência energética, mas têm desempenho limitado sob cargas elevadas. No

segundo cenário, tempos de serviço mais curtos melhoraram o TP e reduziram o DP, exigindo maior controle operacional, enquanto tempos maiores proporcionaram maior estabilidade com menor desempenho. Esses resultados indicam que a escolha da configuração MQTT deve considerar o perfil de carga e os recursos disponíveis, apoiando o planejamento de capacidade em aplicações IoT. Como limitações, destaca-se a ausência de falhas, latência variável e heterogeneidade de dispositivos. Como trabalho futuro, propõe-se a incorporação desses fatores e a realização de uma análise de sensibilidade sistemática, ampliando a robustez e aplicabilidade do modelo em cenários IoT mais realistas e complexos.

Referências

Dizdarević, J., Carpio, F., Jukan, A., and Masip-Bruin, X. (2019). A survey of communication protocols for internet of things and related challenges of fog and cloud computing integration. *ACM Computing Surveys (CSUR)*, 51(6):1–29.

Doshi, R., Inamdar, S., Karmarkar, T., and Wakode, M. (2024). Distributed mqtt broker: A load-balanced redis-based architecture. In *2024 International Conference on Emerging Smart Computing and Informatics (ESCI)*, pages 1–6.

Fiege, L., Muhl, G., and Behnel, S. (2006). On quality-of-service and publish-subscribe. page 20.

Gemirter, C. B., Şenturca, Ç., and Baydere, Ş. (2021). A comparative evaluation of amqp, mqtt and http protocols using real-time public smart city data. In *2021 6th International Conference on Computer Science and Engineering (UBMK)*, pages 542–547. IEEE.

Gruener, S., Koziolek, H., and Rückert, J. (2021). Towards resilient iot messaging: An experience report analyzing mqtt brokers. In *2021 IEEE 18th International Conference on Software Architecture (ICSA)*, pages 69–79.

Hafaiedh, I. B. (2022). Formal models for the verification, performance evaluation, and comparison of iot communication protocols. In *2022 IEEE 21st International Symposium on Network Computing and Applications (NCA)*, volume 21, pages 131–138. IEEE.

Hmissi, F. and Ouni, S. (2022). Td-mqtt: Transparent distributed mqtt brokers for horizontal iot applications. In *2022 IEEE 9th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*, pages 479–486.

Laghari, S. U. A., Li, W., Manickam, S., Nanda, P., Al-Ani, A. K., and Karuppayah, S. (2024). Securing mqtt ecosystem: Exploring vulnerabilities, mitigations, and future trajectories. *IEEE Access*, 12:139273–139289.

Li, Y. and Fujita, S. (2024). A synergistic elixir-eda-mqtt framework for advanced smart transportation systems. *Future Internet*, 16(3):81.

Little, J. D. and Graves, S. C. (2008). Little's law. *Building intuition: insights from basic operations management models and principles*, pages 81–100.

Maciel, P. (2020). *Mercury Tool Manual v5.0 MoDCS Research Group [http://www.modcs.org]*.

Maciel, P., Matos, R., Silva, B., Figueiredo, J., Oliveira, D., Fé, I., Maciel, R., and Dantas, J. (2017). Mercury: Performance and dependability evaluation of systems with exponential, expolynomial, and general distributions. In *2017 IEEE 22nd Pacific Rim international symposium on dependable computing (PRDC)*, pages 50–57. IEEE.

Maciel, P. R. M. (2023). *Performance, reliability, and availability evaluation of computational systems, volume I: performance and background*. Chapman and Hall/CRC.

Mishra, B. and Kertesz, A. (2020). The use of mqtt in m2m and iot systems: A survey. *IEEE Access*, 8:201071–201086.

Mishra, B., Mishra, B., and Kertesz, A. (2021). Stress-testing mqtt brokers: A comparative analysis of performance measurements. *Energies*, 14(18):5817.

Nam, J., Jun, Y., and Choi, M. (2022). High performance iot cloud computing framework using pub/sub techniques. *Applied Sciences*, 12(21):11009.

Nast, M., Golatowski, F., and Timmermann, D. (2023). Design and performance evaluation of a standalone mqtt for sensor networks (mqtt-sn) broker. In *2023 IEEE 19th International Conference on Factory Communication Systems (WFCS)*, pages 1–8.

Putpuek, N., Putpuek, A., and Phawandee, S. (2023). Performance evaluation of opc ua and mqtt for etat smart lab (esl). In *2023 7th International Conference on Information Technology (InCIT)*, pages 17–21.

Raja, P., Kumar, S., Yadav, D. S., and Singh, T. (2023). The internet of things (iot): A review of concepts, technologies, and applications. *International Journal of Information Technology and Communication (IJITC)*, 3(02):21–32.

Rodriguez, L. G. A. and Batista, D. M. (2023). Resource-intensive fuzzing for mqtt brokers: State of the art, performance evaluation, and open issues. *IEEE Networking Letters*, 5(2):100–104.

Seoane, V., Garcia-Rubio, C., Almenares, F., and Campo, C. (2021). Performance evaluation of coap and mqtt with security support for iot environments. *Computer Networks*, 197:108338.

Silva, F. A., Fé, I., and Gonçalves, G. (2021). Stochastic models for performance and cost analysis of a hybrid cloud and fog architecture. *The Journal of Supercomputing*, 77:1537–1561.

Spohn, M. A. (2022). On mqtt scalability in the internet of things: issues, solutions, and future directions. *Journal of Electronics and Electrical Engineering*, pages 4–4.

Vailshery, L. S. (2024). Number of internet of things (iot) connected devices worldwide from 2019 to 2021, with forecasts from 2022 to 2030. Acesso em: 2 jan. 2025.

Yokotani, T., Ohno, S., Mukai, H., and Ishibashi, K. (2021). Iot platform with distributed brokers on mqtt. *International Journal of Future Computer and Communication*, 10(1):7–12.

Zorkany, M., Fahmy, K., and Yahya, A. (2019). Performance evaluation of iot messaging protocol implementation for e-health systems. *International Journal of Advanced Computer Science and Applications*, 10(11).