CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CAMPUS TIMÓTEO

Igor Martins Santos

UMA ABORDAGEM DE AVALIAÇÃO DE DESEMPENHO DE SERVIDORES WEB COM FOCO NO TEMPO DE RESPOSTA E RECURSOS DE HARDWARE

Timóteo

2020

Igor Martins Santos

UMA ABORDAGEM DE AVALIAÇÃO DE DESEMPENHO DE SERVIDORES WEB COM FOCO NO TEMPO DE RESPOSTA E RECURSOS DE HARDWARE

Monografia apresentada à Coordenação de Engenharia de Computação do Campus Timóteo do Centro Federal de Educação Tecnológica de Minas Gerais para obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Adilson Mendes Ricardo

Timóteo

2020

Igor Martins Santos

UMA ABORDAGEM DE AVALIAÇÃO DE DESEMPENHO DE SERVIDORES WEB COM FOCO NO TEMPO DE RESPOSTA E RECURSOS DE HARDWARE

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação do Centro Federal de Educação Tecnológica de Minas Gerais, campus Timóteo, como requisito parcial para obtenção do título de Engenheiro de Computação.

Trabalho aprovado. Timóteo, 11 de Dezembro de 2020:

_____

Prof. Me. Adilson Mendes Ricardo
Orientador

_____

Prof. Dr. Elder de Oliveira Rodrigues
Professor Convidado

_____

Prof. Me. Talles Quintão Pessoa
Professor Convidado

Timóteo
2020

Emitido em 11/12/2020

CÓPIA DE FOLHA DE ASSINATURAS Nº 1/2020 - DCCTM (11.63.05)
(Nº do Documento: 1)

(Nº do Protocolo: NÃO PROTOCOLADO)

(Assinado digitalmente em 11/12/2020 12:30)
ADILSON MENDES RICARDO
PROFESSOR ENS BASICO TECN TECNOLOGICO
DCCTM (11.63.05)
Matrícula: 2849338

(Assinado digitalmente em 11/12/2020 14:23)
ELDER DE OLIVEIRA RODRIGUES
PROFESSOR ENS BASICO TECN TECNOLOGICO
DCCTM (11.63.05)
Matrícula: 1694225

(Assinado digitalmente em 11/12/2020 12:36)
TALLES QUINTAO PESSOA
COORDENADOR - TITULAR
GLABTM (11.63.02.03)
Matrícula: 1552920

Para verificar a autenticidade deste documento entre em https://sig.cefetmg.br/documentos/ informando seu número: **1**, ano: **2020**, tipo: **CÓPIA DE FOLHA DE ASSINATURAS**, data de emissão: **11/12/2020** e o código de verificação: **659e66393f**

Agradecimentos

Dedico este trabalho aos meus pais que estiveram comigo durante toda a trajetória me dando total apoio, suporte e que sempre prezaram pela minha educação.

A todos os meus familiares que acreditam em mim e torcem para o meu sucesso acadêmico e profissional.

Aos meus professores da graduação que se dedicaram em ensinar com excelência.

Ao meu orientador por ter me ajudado e me guiado durante o desenvolvimento deste trabalho.

Por fim, aos meus amigos do CEFET-MG que estiveram sempre presentes durante grande parte desta jornada.

“Os grandes navegadores
devem sua reputação aos temporais e tempestades”.
Epicuro

Resumo

O desenvolvimento tecnológico em diversas áreas, principalmente na *World Wide Web*, traz diversos benefícios como segurança, agilidade, compartilhamento e acessibilidade. Muitos destes serviços são disponibilizados pela rede Internet, o que torna o número de requisições imprevisíveis e altamente variáveis. Por esses e outros fatores, se faz necessário compreender a capacidade e limitações de servidores *Web*, visto que conhecer tais características permite ao administrador garantir a qualidade de serviço da aplicação. Nesta perspectiva, o presente trabalho propõe um procedimento operacional de avaliação de desempenho, que possa tanto avaliar o sistema quanto servir de base para monitorá-lo, com ênfase em recursos de *hardware* e o tempo de resposta da aplicação. Para cumprir e validar o procedimento, foi desenvolvido um cenário *Web*, por meio de máquinas virtuais, com funcionalidades básicas e comuns no mercado, os quais foram definidos como: login, cadastro, consulta e logout. A partir da ferramenta JMeter, foram aplicados testes de rajadas de 100, 200, 300 e 400 usuários sendo o sistema monitorado com a ferramenta Zabbix. Como resultado, foi possível observar características sobre o *hardware* do sistema relacionados aos recursos da CPU, memória RAM, disco rígido, placa de rede e também pode-se ter uma perspectiva do usuário a partir dos tempos de respostas do sistema. Como resultado do estudo de caso, pode-se observar que o disco rígido foi o elemento que mais impactou negativamente o cenário desenvolvido e para que o sistema funcione de forma eficiente, o mesmo pode comportar até 100 usuários em momentos de rajada com base nas metas definidas no projeto. Por fim, este trabalho contribuiu na área de monitoração e análise de desempenho, podendo ser aplicado em sistemas *Web* com diferentes tecnologias.

Palavras-chave: Servidor *Web*, análise de desempenho, monitoração, rajadas HTTP, *Workload*.

Abstract

Technological development in several areas, mainly in the *World Wide Web*, brings several benefits such as security, agility, sharing and accessibility. Many of these services are available over the Internet, which makes the number of requests unpredictable and highly variable. For these and other factors, it is necessary to understand the capacity and limitations of servers *Web*, since knowing these characteristics allows the administrator to guarantee the service quality of the application. In this perspective, the present work proposes an operational performance evaluation procedure, which can both evaluate the system and serve as a basis for monitoring it, with an emphasis on *hardware* resources and the application response time. To comply with and validate the procedure, a *Web* scenario was developed, using virtual machines, with basic and common features in the market, which were defined as: login, registration, consultation and logout. From the JMeter tool, burst tests of 100, 200, 300 and 400 users were applied and the system was monitored with the Zabbix tool. As a result, it was possible to observe characteristics about the *hardware* of the system related to the resources of the CPU, RAM memory, hard disk, network card and it is also possible to have a user perspective from the system response times. As a result of the case study, it can be seen that the hard disk was the element that most negatively impacted the developed scenario and for the system to work efficiently, it can accommodate up to 100 users in burst moments based on the goals defined in the project. Finally, this work contributed in the area of performance monitoring and analysis, and can be applied to *Web* systems with different technologies.

Keywords:Web server, performance analysis, monitoring, HTTP bursts, Workload

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ACM	Association Computing Machi-nery
API	Application Programming Interface
ARPA	Advanced Research Project Agency
ASN.1	Abstact Syntax Notation One
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CERN	European Laboratory for Particle Physics
CPU	Central Processing Unit
DNS	Domain Name System
FTP	File Transfer Protocol
HTTP	HyperText Transfer Protocol
IEEE	Institute of Electrical and Electronic Engineers
ISO	International Organization for Standardization
MBRT	Model Based on the Response Time
MIB	Management Information Base
MIME	Multipurpose Internet Mail
MTBF	Mean Time Between Failures
OID	Object Identifier
OSI	Open System Interconnection
Procempa	Companhia de Processamento de Dados do Município de Porto Alegre
QoS	Quality of Service,
SMI	Structure of Management Information
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
URL	Uniform Resource Locators
UDP	User Datagram Protocol
WWW	World Wide Web

Sumário

1 Introdução

A internet é um meio de comunicação consolidado em nossa sociedade e indispensável para a mesma. Pessoas, empresas, bancos, instituições, entre outros, fazem uso dessa rede de dados para se comunicarem por sistemas *Web*. Assim, é necessário manter esse processo de troca de mensagens sempre em bom estado, evitando perdas de desempenho e mantendo o usuário sempre satisfeito (CENTURION, 2015).

Baseado em Linhares et al. (2014), a internet está em constante crescimento. Segundo Miniwatts Marketing Group (2019) em 2012, 34,3% da população no mundo podia usar esta ferramenta, já em 2019 essa taxa aumentou para 57,3%. Ou seja, houve um aumento de 65,1% em 7 anos.

Devido ao progresso tecnológico em diversas áreas, principalmente na WWW (*World Wide Web*, que é um repositório de dados e serviços que podem ser acessados pela rede internet), vários serviços sobre a transmissão de dados têm sido criados. Como resultado desse progresso, as redes de computadores permitem que organizações com centenas de escritórios geograficamente distantes, possam compartilhar arquivos, armazenar dados, trocar informações, entre outros (TANENBAUM et al., 2011; COSTA et al., 2009).

A alocação de serviços criados dentro das empresas trazem diversos benefícios como segurança, agilidade, compartilhamento e acessibilidade. Lojas virtuais, sistemas bancários, caixas de supermercado, sistemas EAD (Ensino a Distância) e diversos outros, utilizam de sistemas *Web* que muitas vezes desenvolvem um papel crítico dentro da organização. A falta destes serviços pode ocasionar a parada de toda uma linha de produção, levando a graves prejuízos à empresa (COSTA et al., 2009).

Um dos focos atuais do mercado é a computação em nuvem. A grande diferença é que as empresas podem contratar o serviço e a infraestrutura, ou seja, a entrega sob demanda de poder computacional, armazenamento de banco de dados, aplicativos e outros recursos de TI. Assim, o usuário não terá que se preocupar com diversos fatores como escalabilidade, refrigeração, manutenção, equipamentos obsoletos, monitoração entre outros. O que traz grandes vantagens para pequenas ou grandes empresas (REHMAN et al., 2015; DATT; GOEL; GUPTA, 2015).

Outro fato importante, citado por Valentini e Gaspary (2003) e MICHELON e HILD (2009), é que essa disponibilidade de serviços oferecidos pela rede aumenta à medida que cresce a capacidade de processar, colher e distribuir informações. Conforme tais sistemas crescem, principalmente os que comportam servidores que oferecem serviços pela WWW, novos atributos são acrescentados e/ou o número de usuários aumenta. Tais situações podem acarretar sobrecarga.

Diante da natureza imprevisível e altamente variável da WWW, os servidores podem ser atingidos por diversos fenômenos prejudiciais ao seu desempenho. Dentre eles destaca-se

a existência das chamadas de "rajadas". O comportamento da taxa de requisições na forma de rajadas é uma característica usual de cargas de trabalho encontradas em serviços e aplicações *Web* (CENTURION et al., 2012).

A rajada é caracterizada pelo aumento inesperado de requisições ao servidor. O termo surgiu por volta de 1970 e 1980 onde pesquisadores o usavam para indicar o fenômeno que se caracteriza quando a taxa de chegada de requisições era altamente instável. Basicamente, o número de requisições aumentava repentinamente (CENTURION et al., 2012).

Servidores *Web* são necessários para realizar milhões de pedidos de transações por dia em uma qualidade aceitável de nível de serviço em termos de tempo de resposta do cliente e taxa de transferência servidor. Portanto, uma compreensão completa das capacidades de desempenho e limitações de servidores *Web* é crítica para a garantia do QoS (*Quality of Service*, se refere a capacidade de melhorar os serviços trafegados na rede) da aplicação. (ZHANG; ZHU, 2014).

1.1 Justificativa e Problema

Há a necessidade de métodos e ferramentas de teste que possam ajudar de maneira prática a avaliar a escalabilidade e robustez de uma aplicação sob condições negativas e monitorar de forma efetiva para identificar tais fenômenos. Infelizmente, devido à natureza orientada por sessão das cargas de trabalho, não há previsibilidade da variação de requisições, o que torna essa tarefa não trivial (CASALE et al., 2012).

À medida que os sistemas computacionais se desenvolvem, existe a necessidade de planejar, implementar e monitorar. Esses sistemas, que normalmente são baseados na arquitetura cliente/servidor, podem ser complexos devido ao tamanho da rede e a grande heterogeneidade de serviços ofertados (BENICIO, 2015).

Para avaliar de forma adequada os servidores *Web*, que são computadores que hospedam um ou mais sites/aplicações na internet, é importante fazer a análise de desempenho, avaliação e planejamento. Um modelo de avaliação oferece diversos benefícios, como controlar sobrecarga, orientar a alocação de recursos, comparar diferentes plataformas de hardware e sistemas operacionais entre outros.

Na literatura há diversos trabalhos que podem ser divididos em dois tipos, os baseado em análise/simulação e baseado em medição. Tais trabalhos têm focado em avaliar servidores *Web* e trazem diversos elementos diferentes para tentar prever fenômenos negativos. Alguns destes são: tamanho do *buffer*, uso de CPU, uso de memória RAM, tamanho do arquivo enviado, tempo de respostas entre outros (Xianghua Xu et al., 2013).

Neste contexto, dentre as possibilidades de softwares disponíveis no mercado, a etapa de construção do cenário de teste pode ser uma tarefa difícil, pois envolve a análise de diversos fatores externos e internos do sistema. Com base nos objetivos, são escolhidas as ferramentas que farão parte do projeto visando tanto qualidade dos resultados obtidos, quanto à aplicabilidade e facilidade no manuseio das mesmas (VALDRICH; MISAGHI, 2015).

Desta forma, os trabalhos desenvolvidos, mesmo com foco apenas nos resultados, podem auxiliar no desenvolvimento da etapa inicial do projeto, ou seja, na construção da análises de desempenho, auxiliado não só na escolhas do *software*, mas também na escolha das métricas e como utilizá-la para reconhecer gargalos no sistema.

Segundo SILVA (2015), avaliar o desempenho de qualquer objeto é uma tarefa bastante importante para qualquer área, principalmente em servidores *Web*. Fica evidente a necessidade do monitoramento de recursos e desempenho, sendo que cada vez mais serviços são empregados a tais máquinas e que a grande demanda de requisições pode saturar até sistemas com um alto poder computacional.

Assim, é de suma importância o uso da monitoração e da análise de desempenho em servidores *Web*, visto que, avaliar o desempenho analisando métricas relacionadas recursos hardware pode ser uma forma eficientemente de prever condições de gargalo em aplicações Cliente/Servidor em momentos de rajada. Ademais, o uso de metodologias para a montagem do ambiente pode auxiliar outros projetos tanto na análise dos resultados, quanto na montagem ambiente da avaliação de desempenho.

Com base nos argumentos, o presente trabalho procura aplicar um procedimento operacional de avaliação de desempenho, o qual responda às seguintes perguntas: Quais são os elementos hardware, com base na bibliografia, que podem ser monitorados remotamente e que são sensíveis aos momentos de rajada em servidores *Web* ? Sob o ponto de vista do usuário, como esses elementos de hardware impactam no desempenho final de uma aplicação cliente/servidor ?

1.2 Objetivo

Este trabalho tem como objetivo criar um procedimento operacional de avaliação de desempenho e monitoração para o sistema operacional *Windows Server* 2019, simulando o fenômeno de rajada HTTP visando avaliar a aplicabilidade dos elementos de hardware sensíveis a tal e o impacto sobre o cliente final.

1.3 Objetivo específicos

- Aplicar metodologia de avaliação de desempenho.
- Simular o fenômeno de rajadas HTTP em um servidor *Web*.
- Analisar uma ferramenta de monitoração atual para o processo de coleta de dados sobre o servidor.
- Coletar de forma remota, interpretar e analisar os dados dos elementos de hardware, que são sensíveis ao uso de recursos físicos da máquina.

2 Revisão bibliográfica

Este capítulo apresenta os estudos teóricos que foram usados para o embasamento desta pesquisa.

2.1 Gerência de Redes

Segundo Forouzan e Mosharraf (2013), década de 60, os mainframes (Computadores de grande porte e de custo elevado) eram utilizados em grandes organizações de pesquisa. Porém, esses grandes computadores não eram capazes de se comunicar caso não fossem do mesmo fabricante. A *Advanced Research Project Agency* (ARPA) estava interessada em encontrar um meio de realizar essa troca de dados entre os dispositivos.

No período de 1967, foi apresentado no encontro da *Association Computing Machinery* (ACM) a ARPANET, que era uma rede pequena de computadores. A principal ideia era padronizar o meio de comunicação fazendo com que os diversos computadores da época pudessem se comunicar. Em 1969, a ARPANET foi aplicada, interligando várias universidades (MACEDO et al., 2018).

A evolução crescente dos meios de comunicação e tecnologias, proporcionou desenvolvimento de alguns serviços como e-mail, FTP (*File Transfer Protocol*), que é um tipo de conexão que permite a troca de arquivos entre dois computadores conectados à internet) e DNS (*Domain Name System*, responsável por localizar e traduzir para números IP o endereço do site que foi digitado no navegador), sendo estes a base de todos os serviços atuais criados(FOROUZAN; MOSHARRAF, 2013).

Na década de 90, houve o crescimento do acesso à rede internet, onde a mesma ganhou maior popularidade. Com isso, a mesma se tornou mais acessível, o que levou as redes de computadores para um novo patamar, assim sendo até hoje, uma das áreas de grande importância e evolução tecnológica (MACEDO et al., 2018).

Neste contexto, o gerenciamento de rede pode ser entendido como a atividade de testar, monitorar, configurar e resolver problemas de rede. Assim evitando que os serviços oferecidos a seus usuários parem de funcionar. Essas tarefas demandam operações regulares sobre a rede. Para realizar tais tarefas de controle e monitoramento, deve-se utilizar recursos vindos de hardwares e softwares a fim de automatizar esses processos. (FOROUZAN; MOSHARRAF, 2013; PINHEIRO, 2002).

De acordo com Pinheiro (2002), Kurose e Ross (2010), foram estabelecidos pela *International Organization for Standardization* (ISO), três modelos: organizacional, informacional e funcional. O modelo organizacional, que descreve a forma pela qual a gerência pode ser distribuída em unidade administrativa que possuem a mesma característica e/ou funções, ou seja, estabelece uma hierarquia entre os sistemas de gerenciamento, separando ambientes a serem gerenciados em vários domínios.

Já o modelo informacional provê a base para a definição de objetos gerenciados e suas relações, classes, atributos, ações e nomes. Definindo assim os objetos de gerência e as suas operações e relações. Para isso, foi necessário a criação da MIB (*Management Information Base*), que será explicado com maior profundidade posteriormente neste capítulo seção MIB 2.3.

E por fim, o modelo funcional descreve as áreas funcionais e seus relacionamentos: gerência de falhas, gerenciamento de conta, gerência de configuração, gerência de desempenho e gerência de segurança (SILVA; OLIVEIRA, 2014; SANTOS et al., 2015).

2.1.1 Gerenciamento de falhas

Para manter o funcionamento adequado de uma rede complexa, deve-se ter cuidado com os sistemas como um todo, mantendo cada componente essencial em funcionamento adequado. Quando ocorre uma falha, é importante verificar o mais rápido possível onde está a mesma e resolvê-la.

Em relação aos usuários os mesmos esperam uma resolução de problemas rápida e confiável. Quando essas interrupções infrequentes ocorrem, o próprio geralmente espera receber uma notificação imediata e que o problema seja corrigido em um curto período de tempo. Para fornecer esse nível de solução, o sistema necessita de funções de detecção e gerenciamento de diagnóstico muito rápidas e confiáveis.

Assim, gerenciamento de falhas é definido como um conjunto de elementos que facilite a reprodução e entendimento de logs de erros, fornece recursos para receber e registrar notificação de erros e atuar em função destas, rastrear problemas, executar sequências de testes de diagnóstico e corrigir problemas com objetivo de assegurar a operação contínua da rede (SANTOS et al., 2015).

> O objetivo do gerenciamento de falhas é registrar, detectar e reagir às condições de falha da rede. A linha divisória entre gerenciamento de falha e gerenciamento de desempenho é bastante indefinida. Considera-se o gerenciamento de falhas como o tratamento imediato de falhas transitórias da rede (por exemplo, interrupção de serviço em enlaces, hospedeiros, ou em hardware e software de roteadores), enquanto o gerenciamento de desempenho adota uma abordagem de longo prazo em relação ao desempenho da rede em face de demandas variáveis de tráfego e falhas ocasionais na rede. Como acontece no gerenciamento de desempenho, o SNMP (Simple Network Management Protocol, protocolo usado para monitorar elementos de rede) tem um papel fundamental no gerenciamento de falhas. (KUROSE; ROSS, 2010, p. 555).

2.1.2 Gerenciamento de conta

O gerenciamento de contas permite que sejam estabelecidos níveis para usuários, habilitando o uso de objetos gerenciados e sendo possível avaliar os custos a serem identificados para o uso desses objetos gerenciados. Em muitas redes corporativas, divisões individuais ou centros de custo, ou até mesmo contas de projetos individuais, são cobrados pelo uso de serviços de rede. Esses são procedimentos contábeis internos, que são importantes para os usuários participantes (LEME, 2016; KUROSE; ROSS, 2010).

O gerenciador de rede precisa ser capaz de identificar o uso de recursos de rede utilizado pela classe de usuário por vários motivos, incluindo(LEME, 2016):

- Um usuário ou grupo de usuários pode estar abusando de seus privilégios de acesso e sobrecarregar a rede.
- Os usuários podem estar fazendo uso ineficiente da rede e o gerenciador de rede pode ajudar na administração de mudanças de procedimentos para melhorar o desempenho.
- O gerente de rede está em uma posição melhor para planejar o crescimento da rede se a atividade do usuário é conhecida em detalhes suficientes.

É importante garantir a segurança das informações da organização, por isso o serviço de rede deve ser capaz de autenticar cada usuário para que o mesmo possa manipular informações baseada na sua função dentro da organização.

2.1.3 Gerenciamento de configuração

As redes de comunicação de dados modernas são compostas de componentes e subsistemas lógicos como por exemplo, um *driver* de dispositivo em um sistema operacional, que pode ser configurado para executar muitos aplicativos diferentes.

Por exemplo, um mesmo equipamento, pode ser configurado para atuar como um roteador ou como um *switch layer 2*, que é um dispositivo que funciona na camada de enlace de dados do modelo OSI (*Open System Interconnection*, modelo que define padronização para protocolos de comunicação entre os mais diversos sistemas em uma rede). Uma vez decidido como um dispositivo deve ser usado, o gerenciador de configuração escolhe o software apropriado e o conjunto de atributos e valores para esse dispositivo.

O gerenciamento de configurações se preocupa em conhecer dispositivos que fazem parte da rede administrada. Permitindo manter, adicionar e atualizar os relacionamentos entre os elementos e os estado dos componentes durante a operação da rede, ou seja, permite administrar os elementos e ter uma visão geral de suas configurações de hardware e software (KUROSE; ROSS, 2010).

> Gerenciamento de configuração é o conjunto de facilidades que exerce o controle sobre o que está na rede: identifica, coleta e provê informações sobre o status dos recursos da rede de forma que torne possível a qualquer instante conhecer o que está operacional, em uso ou simplesmente disponível para utilização em caso de emergência(SANTOS et al., 2015, p. 555).

Desta forma, o gerente de rede precisa da capacidade de conhecer, alterar e configurar a conectividade dos componentes de rede quando as necessidades dos usuários mudarem, ou em resposta à avaliação de desempenho, ou no suporte a atualizações de rede, ou em recuperação de falhas ou verificações de segurança.

2.1.4 Gerenciamento de desempenho

Redes de comunicação de dados são compostas de variados componentes, que devem intercomunicar e compartilhar dados e recursos. Em alguns casos, é fundamental para a eficácia de uma aplicação que a comunicação através da rede apresente níveis em conformidade a certos limites de desempenho.

A gestão de desempenho de uma rede de computadores atinge duas amplas categorias funcionais: monitoramento e controle. Monitoramento é a função que identifica atividades na rede. A função de controle permite que o gerenciamento de desempenho faça ajustes para melhorar o desempenho da rede. Alguns dos problemas de desempenho que preocupam o gerente de redes são os seguintes (LEME, 2016; SANTOS et al., 2015):

- Qual é o nível de utilização máxima?
- Existe tráfego excessivo?
- A taxa de transferência foi reduzida a níveis inaceitáveis?
- Existem gargalos?
- O tempo de resposta está aumentando?

Para lidar com essas preocupações, o gerente de rede deve se concentrar em um conjunto de recursos a serem monitorados para avaliar os níveis de desempenho. Isso inclui associar métricas e valores apropriados a recursos de rede relevantes como indicadores de diferentes níveis de desempenho (SANTOS et al., 2015).

O gerenciamento de desempenho, portanto, deve monitorar muitos recursos para fornecer informações na determinação do nível operacional da rede. Coletando essas informações, será possível avaliar e, em seguida, usar a análise resultante como *feedback*. Assim o gerente de rede pode tornar-se mais apto a reconhecer situações indicativas de degradação de desempenho presente na rede e não deixar que o desempenho negativo interfira na qualidade de serviço(LEME, 2016).

2.1.5 Gerenciamento de segurança

A gestão de segurança está preocupada com a geração, distribuição e armazenamento de chaves de criptografia. Senhas e outras informações de autorização ou controle de acesso devem ser mantidas e distribuídas. Gestão de segurança também está preocupada com o monitoramento e controle de informações de acesso a redes de computadores e acesso a todos os dispositivos da rede (SANTOS et al., 2015; KUROSE; ROSS, 2010).

O gerenciamento de segurança é associado com a coleta, armazenamento e exame de registros de auditoria e registros de segurança, bem como com a ativação e desativação dessas instalações de registro. Esses aspectos da segurança são essenciais para garantir a integridade, autenticidade, disponibilidade e confidencialidade das informações e dos recursos (LEME, 2016):

- Integridade: garante que a informação manipulada mantenha todas as características originais estabelecidas pelo proprietário da informação.
- Autenticidade: garante que a informação é proveniente do remetente e que não será alterada ao longo de um processo.
- Disponibilidade: garante que a informação esteja sempre disponível para o uso dos usuários autorizados.
- Confidencialidade: propriedade que limita o acesso a informação aos usuários autorizados pelo proprietário da informação.

2.2 Estrutura de gerenciamento

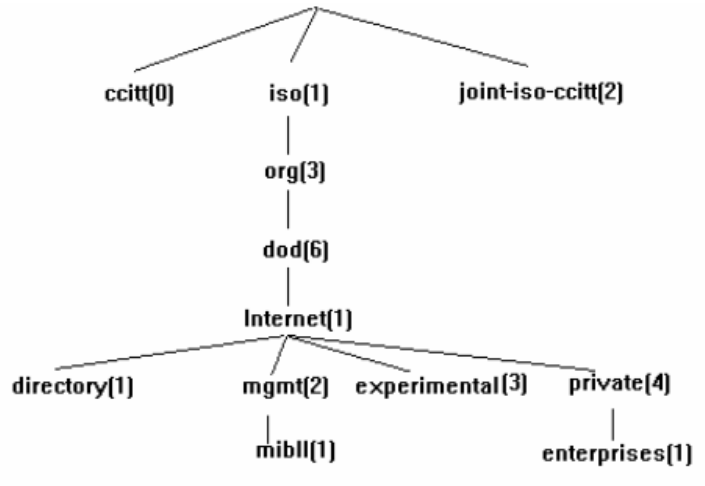
Segundo (KUROSE; ROSS, 2010, p. 560) o gerenciamento possui quatro estruturas essenciais. A primeira é o SMI (*Structure of Management Information*) que é baseada na ASN.1 (*Abstact Syntax Notation One*), é a linguagem que define as regras utilizadas para a troca de informação. O segundo é a MIB (*Management Information Base*) que é uma base de dados formada por objetos monitorados como por exemplo CPU (Central Processing Unit), placa de rede, *interfaces*. O protocolo SNMP é utilizado para troca de informação entre agente e gerente. E por último a capacidade de administração e manipulação de tais estruturas a fim de garantir a eficiência e segurança dos itens monitorados.

2.3 MIB - *Management Information Base*

A MIB é definida como um conjunto de objetos gerenciáveis que traz diversas informações necessárias para a monitorar a rede. Os objetos gerenciáveis são definidos como a representação abstrata de um recurso ou característica gerenciável de um determinado dispositivo, assim os recursos da rede devem ser mapeados e modelados nos dispositivos. Esses objetos contém dados do estado atual, que podem ser lidos e/ou alterados (DIAS; JR, 2002; ARANTES et al., 2012).

A MIB possui uma arquitetura baseada em uma estrutura em árvore onde cada nó folha contém os objetos de gerência de um determinado equipamento. A mesma define para cada nó de determinado nível da árvore, um inteiro não negativo único, chamado de OID (*Object Identifier*). Com isso, todos os objetos definidos em todos os padrões oficiais podem ser exclusivamente identificados. Para acessar ou localizar um determinado objeto gerenciável, basta identificar o endereço de IP do dispositivo, junto ao identificador do objeto na árvore MIB (OID) (CONTESSA; POLINA, 2010). A Figura 1 demonstra a estrutura da MIB:

Figura 1 – Parte da estrutura hierárquica da MIB



Fonte: Arantes et al. (2012)

O OID de um nodo da árvore é representado por um conjunto de identificadores, que é composto pelo identificador de seu pai mais seu próprio. No entanto, o uso de número nos OID's pode dificultar a compreensão dos nodos, por isso é possível substituir tal valor por um nome (OID Name), podendo ser usado em conjunto. Como exemplo o OID 1.3.6.1.2.1.1.3 pode ser representado pelo OID Name "system", ou seja, o OID 1.3.6.1.2.1.1.3 pode ser representado como system.3 (CONTESSA; POLINA, 2010).

Ainda no trabalho de Arantes et al. (2012) são definidos 4 tipos de MIB's: MIBI, MIBII, MIB experimental e MIB privada. As MIBI e MIBII sendo a segunda evolução da primeira, fornecem dados genéricos, que podem ser aplicados na grande maioria dos dispositivos de rede. A partir desta MIB's é possível obter dados de número de pacotes de dados, portas UDP (User Datagram Protocol, protocolo da camada de transporte, permite que a aplicação envie um datagrama a um destino, todavia sem garantir a entrega do mesmo) e TCP (Transmission Control Protocol, protocolo da camada de transporte que permite o envio de dados de forma segura) utilizada, status de interface entre outros.

As MIB'S experimentais como o próprio nome diz, são as que estão em fase de teste e que podem ser adicionadas futuramente ao padrão. A MIB privada é aquela que seus componentes fornecem informações específicas dos equipamentos gerenciados, por exemplo, a configuração, colisões e também é possível reinicializar, desabilitar uma ou mais portas de um roteador (ARANTES et al., 2012).

2.4 SNMP

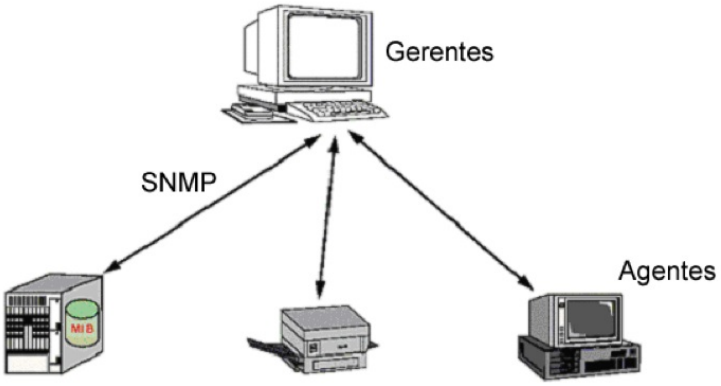
Segundo Dias e Jr (2002), o *Simple Network Management Protocol* (SNMP) foi desenvolvido pelo *Internet Engineering Task Force* – IETF e lançado por volta de 1988, com objetivo de atender a necessidade de padronização de um protocolo para gerenciar os dispositivos de rede. O SNMP é um protocolo em nível de aplicação, possui uma arquitetura baseada em gerente e agente. Os agentes são baseados no protocolo TCP/IP e para a troca de dados é

utilizado o protocolo UDP (User Datagram Protocol).

Os objetos que podem ser monitorados pelo SNMP são definidos na árvore de objetos MIB, as MIBI e MIBII, as quais especificam e padronizam os dados que podem ser requisitado nos dispositivos em geral, a MIB privada pode trazer objetos variado de empresa ou dispositivo. Os comando são baseados em busca e alteração, esse mecanismo possibilita operações de alteração de valores de um objeto ou de obtenção de algum parâmetro, em um dispositivo (MAURO; SCHMIDT, 2001; ARANTES et al., 2012).

O agente SNMP é um processo executado no dispositivo a ser monitorado, ele tem a função de receber as requisições e encaminhar a resposta ao gerente, e além, pode ser configurado para enviar informações automaticamente. O gerente é um programa executado em uma nó servidor que faz as requisições e tem capacidade de receber as respostas, de um ou mais agentes na rede. A Figura 2 representa a comunicação entre gerente e agentes:

Figura 2 – Gerente e Agente



Fonte: (VIEIRA, 2010)

Como já citado o SNMP é um protocolo de requisição/resposta, assim são definidas operações de leitura, escrita e notificação de condições específicas (traps). O Protocol Data Unit (PDU) é o formato de mensagem que os gerentes e agentes usam para enviar e receber informações. Existe um formato de PDU padrão para cada uma das seguintes operações SNMP: (CONTESSA; POLINA, 2010; CASE et al., 1990; VIEIRA, 2010)

- *Get*;
- *Get-next*;
- *Get-bulk (SNMPv2 and SNMPv3)*;
- *Set*;
- *Get-response*;
- *Trap*;
- *Notification (SNMPv2 and SNMPv3)*;
- *Inform (SNMPv2 and SNMPv3)*;

- *Report (SNMPv2 and SNMPv3)*;

2.5 Avaliação de desempenho

Para sistemas computacionais, o desempenho pode ser correspondente a uma avaliação da quantidade de serviço prestado ao longo de um período de tempo. Mesmo não sendo um parâmetro confiável, tais sistemas computacionais também podem ser avaliados pelos usuário que o utiliza, então, dependendo do grau de contentamento do grupo, se faz necessário realizar uma avaliação minuciosa, à procura de gargalos (Moraes Junior, 2016).

Um exemplo que podemos citar são sistemas de bancos de dados, os mesmos recebem requisições de diversos usuários para realizar algum tipo de serviço, como uma consulta em uma tabela, inserção de dados em um registro, alterar tabelas e etc. Outro exemplo poderia ser um sistema de redes de computadores, por onde trafegam pacotes(JOHNSON; MARGALHO, 2014).

De forma análoga, tais sistemas recebem requisições (pedidos, jobs, pacotes) de clientes para executar alguma tarefa, ou seja, algum tipo de serviço. Para se avaliar tais serviços deve-se considerar diversas características, como por exemplo o tipo de cliente, o tipo de requisição, o tipo de serviço entre outros(JOHNSON; MARGALHO, 2014). A Tabela 1 mostra exemplos de avaliação em sistemas computacionais:

Tabela 1 – Exemplos de avaliações em sistemas

Tipo de sistema	**Medição**	**O que comparar**
Software cliente	Tempo de inicialização, requisições atendidas por minuto.	Uso do *software* em diferentes computadores.
Redes de computadores	Utilização de banda, vazão de pacotes.	Link de rede mais rápido.
Banco de dados	Consultas atendidas por minuto, tempo de atendimento de consulta.	Qual melhor servidor de banco de dados.
Sistema de arquivos e memória	Taxa de transferência de blocos/páginas.	Melhor sistema operacional para operações físicas.

Fonte: Adaptado de Moraes Junior (2016)

Apesar de suma importância, a avaliação de sistemas podem ser ignoradas pelo administrador de rede, devido a falta de conhecimento de ferramentas de avaliação ou a dificuldade em executá-las. Tal descuido pode levar a grandes perdas econômicas. Dependendo do cenário, é possível utilizar diversas técnicas, como por exemplo *benchmarks*, prototipação, coleta de dados e etc, ou modelagem (solução analítica ou por simulação)(Moraes Junior, 2016).

2.5.1 Técnicas de avaliação

Para avaliar o desempenho, deve-se realizar a coleta de dados, referentes aos parâmetros que se desejava analisar. Para isso faz-se necessário a utilização de técnicas para a obtenção deste valores. Segundo Johnson e Margalho (2014), existem basicamente três técnicas de desempenho: modelagem, simulação e medição (aferição).

A escolha da técnica pode variar de acordo com o sistema. Em cenários em que o mesmo não foi desenvolvido ou é inviável a realização de testes em produção é possível usar simulação ou modelagem. Tais técnicas permitem uma avaliação em diferentes cenários, possibilitando a comparação entre os mesmos. A partir disso o gestor poderá recriar fenômenos que o sistema real sofreria e por meio de análises, terá base para configurar o sistemas de formas mais vantajosas, e avaliar/identificar os possíveis problemas ao implementar o sistema (JOHNSON; MARGALHO, 2014; Moraes Junior, 2016).

Para uma maior confiança, é interessante utilizar mais de uma técnica de avaliação de desempenho. Pode-se obter as seguintes combinações:

Modelagem + Simulação
Modelagem + Medição
Simulação + Simulação
Modelagem + Simulação + Medição

2.5.1.1 Modelagem

Para grandes organizações como indústrias, centros de pesquisa não é interessante construir um sistema e só após avaliar seu desempenho. Se for possível realizar tais testes antes mesmo do sistema ser desenvolvido, tal iniciativa pode representar uma redução considerável nos custos. Para esse tipo de situação é desejado que se modele o sistema antes mesmo de existir (JOHNSON; MARGALHO, 2014; Moraes Junior, 2016).

Um modelo é uma representação de um determinado sistemas, que abstrai grande parte de suas características, porém evidências as principais. Esses modelos sofrem alterações em função do tempo. Os modelos podem ser classificados como discretos ou contínuos. Como em sistemas computacionais as mudanças de estado são discretas, os modelos são discretos também (JOHNSON; MARGALHO, 2014; Moraes Junior, 2016).

Podemos classificar algumas das principais técnicas de modelagem que são: teoria das filas, análise operacional de filas, análise quantitativa de filas de processos estocásticos, redes de petri, entre outros (Moraes Junior, 2016).

2.5.1.2 Simulação

Na computação, o emprego de uma simulação está ligado ao uso de um programa que permite obter respostas de algum fenômeno de um sistema dinâmico (JOHNSON; MARGALHO, 2014).

A técnica de simulação é largamente usada em sistemas computacionais devido a linha de aprendizado de uso. A mesma é usada para prever o desempenho de um sistema inexistente. Por exemplo, em cenários que a necessidade de alteração de hardware, como por exemplo a CPU, existem diversas técnicas baratas para se realizar testes de desempenho antes e após as modificações (Moraes Junior, 2016).

Porém, um problema em relação a simulação é a geração de modelos que são falhos, onde as respostas não são corretas. Por isso é necessário usar simuladores confiáveis, que já tenham sido testados pela comunidade científica ou profissional (JOHNSON; MARGALHO, 2014; Moraes Junior, 2016).

2.5.1.3 Experimento ou Aferição ou Monitoramento

É possível realizar quando um sistema existe, através de experimentos ou aferição se coleta dados, aplicando diferentes cargas de trabalho sobre o sistema. Dependendo do mesmo, diversos tipos de dados podem ser coletados em situações diferentes (Moraes Junior, 2016).

Um sistema de monitoração pode capturar informações de desempenho para análise posterior ou em tempo real. Como exemplo, uma rede de computadores pode possuir um sistema de monitoração que colete dados de tráfego ou até mesmo informações de hardware de dispositivos empregados na rede (JOHNSON; MARGALHO, 2014).

Outra forma de estudo de experimentação é através do uso de benchmarking, que são softwares usados para testar o desempenho de sistemas aplicando carga e coletando dados. Por exemplo, pode-se avaliar a CPU de um computador, aplicando um estresse sobre o mesmo e recolhendo dados. Geralmente tais testes são usados para avaliar processadores, sistemas de entrada/saída, redes de computadores e banco de dados (JOHNSON; MARGALHO, 2014).

2.6 WWW - *World Wide Web*

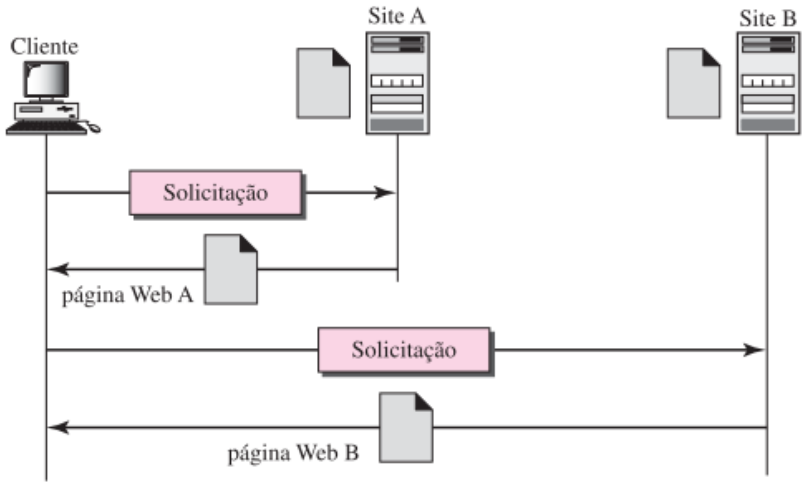
Antes da década de 1990, a Internet era usada principalmente por pesquisadores, acadêmicos e estudantes para a transferência de arquivos, trocas de notícias e correios eletrônicos. Nesta época a internet ainda não era conhecida fora desta comunidade. A partir de 1990, uma nova aplicação foi criada, a *World Wide Web*, que foi iniciado pelo Cern (*European Laboratory for Particle Physics*) Laboratório europeu de estudos das partículas elementares (KUROSE; ROSS, 2010; FOROUZAN, 2008).

Esta nova aplicação é um repositório de informações interligadas por diversos pontos espalhados ao redor do mundo. A WWW trouxe uma combinação de flexibilidade, facilidade de envio de recursos, o que facilitou seu uso (FOROUZAN, 2008). Alguns dos pontos que podem ter atraído tantos usuários segundo (KUROSE; ROSS, 2010), é o fato de que ela funciona por demanda, ou seja, os usuários podem solicitar as informações de seu interesse quando quiserem, ao contrário de outros meios de comunicação como a TV e o rádio. Ademais, a mesma permite, de forma fácil, disponibilizar informações na *Web*, qualquer usuário pode ser capaz de distribuir informações por um custo baixo. Além disso, as páginas *Web* trazem

interações a partir de gráficos, formulários, arquivos multimídia entre outros, tornando páginas e sites mais atraentes aos usuários.

A arquitetura da WWW é baseada na cliente/servidor, onde o cliente a partir de um browser (aplicação que interpreta e exibe um documento *Web*) pode acessar dados ou serviços que estão hospedados em um servidor ou diversos servidores distribuídos. A Figura 3 representa a arquitetura da WWW:

Figura 3 – Arquitetura WWW



Fonte: (FOROUZAN, 2008)

Essa estrutura permite que um documento possa referenciar outros documentos pela WWW, criando uma estrutura de interligação entre os documentos que podem estar hospedados em máquinas distribuindo na internet. Tais páginas são referenciadas utilizando o URL (*Uniform Resource Locators*), que é um endereço virtual com um caminho que indica onde está o que o usuário procura (JUNIOR, 2008; FOROUZAN, 2008).

O URL é dividido em 5 partes: protocolo utilizado, o nome do servidor acessado através do DNS (*Domain Name System* ou sistema de nomes de domínios, estes que são os responsáveis por localizar e traduzir para números IP os endereços dos sites que digitamos nos navegadores), a porta pela qual o servidor recebe as conexões (padrão porta 80), o diretório onde se está armazenado os arquivos e o nome do arquivo. Na Figura 4 identifica-se esquema do protocolo URL:

Figura 4 – Protocolo URL

Fonte: (JUNIOR, 2008)

2.7 HTTP — HyperText Transfer Protocol

O *Hypertext Transfer Protocol*, mais comumente chamado de HTTP, é um protocolo em nível de aplicação usado principalmente para troca informações na *Web*. O HTTP é implementado em dois programas, o cliente e o servidor. Estes dois se comunicam por meio de trocas de mensagens HTTP. Ele define as estruturas das mensagens que o cliente pode enviar ao servidor e as respostas que serão devolvidas (KUROSE; ROSS, 2010; TANENBAUM et al., 2011).

O mesmo pode ser dito como a junção de dois protocolos, o FTP (File Transfer Protocol, permite a troca de arquivos entre dois computadores conectados à internet), pois permite a transferência de arquivos e utiliza o protocolo TCP. Porém ele é muito mais simples pois usa apenas uma conexão TCP, onde somente dados são transferidos entre o cliente e o servidor, enquanto o FTP cria uma outra conexão para controle(FOROUZAN, 2008).

O outro protocolo similar é o SMTP, pois os dados que são trocados são semelhantes e o cabeçalho das mensagens são encapsuladas em um formato parecido com o MIME (Multipurpose Internet Mail), que permite inserir diferentes tipos de dados como imagens, áudio e textos em caracteres diferentes (FOROUZAN, 2008). A figura 5 representa o comportamento de requisição-resposta do HTTP.

Figura 5 – Comportamento de requisição-resposta do HTTP

Fonte: (KUROSE; ROSS, 2010)

Existem duas versões do HTTP, na 1.0 a conexão é não persistente, pois após a conexão ser estabelecida, apenas uma única solicitação é enviada ao servidor, logo o cliente recebe uma única resposta, finalizando a conexão TCP. Porém, essa versão só é vantajosa quando as páginas da *Web* consistem de apenas textos HTML. Ao se passar alguns anos, as página da *Web* passaram a conter grandes números de ícones, imagens e outros tipos de arquivos. nesta situação para N imagens e/ou arquivos distintos, a conexão era iniciada N vezes, o que causa um *overhead* nos servidores. Isso devido ao procedimento de inicialização

das conexões e pelo número de *buffers* diferentes que o servidor precisava para manter as conexões(FOROUZAN, 2008; TANENBAUM et al., 2011).

Isso levou ao lançamento do HTTP 1.1, que admite conexão persistente, onde é possível estabelecer vários ciclos de requisições/respostas em uma mesma conexão, o que minimizou o *overhead* relativo ao HTTP 1.0. Também possibilitou transportar as solicitações por pipeline, ou seja, enviar a solicitação 2 antes de receber a resposta da solicitação 1. (FOROUZAN, 2008; TANENBAUM et al., 2011).

2.8 Servidores *Web*

Segundo (MESSIAS, 2007; JUNIOR, 2008), um servidor *web* pode ser definido como um processo em um laço de repetição infinito à espera de requisições de clientes. Durante o procedimento de atendimento, o servidor pode sofrer diversos atrasos causados por fatores *hardware*, *software* e rede como: leitura de disco, congestionamento dos roteadores da Internet, espera pela transferência de dados pela rede, escalonamento de processos, atrasos relacionados ao servidores de banco de dados, entre outros. Assim, os servidores devem ser projetados para atender o máximo de requisições possíveis.

Abaixo serão abordados algumas estratégias usadas em servidores *web* baseadas nos trabalhos de Messias (2007), Junior (2008):

- **Servidor Iterativo:**

  A arquitetura mais simples de se implementar um servidor, funciona de forma sequencial, ou seja, atende às requisições uma de cada vez respeitando a ordem: o primeiro a entrar é o primeiro a ser atendido. Esse método é muito ineficiente quando se refere a um grande número de requisições, servindo apenas para fins didáticos.

- **Processo por Requisição:**

  Essa arquitetura possui dois tipos de processos, o principal que é chamado de *dispacher*, tal processo aguarda as requisições e fica responsável por criar o outro tipo de processos que é *worker*, este fica responsável por executar a tarefa delegada, após a finalização da tarefa o processo *worker* é eliminado.

  Essa estratégia não funciona bem em servidores que trabalham com grandes cargas de trabalho, pois existe um custo computacional ao se criar novos processo, logo em picos de requisições pode haver uma queda no desempenho.

- ***Pool* de Processos:**

  A ideia de se usar um pool de processos é para tentar aproveitar a simplicidade da abordagem anterior, porém minimizar suas desvantagens. Essa arquitetura cria os processos e os deixa de prontidão para atender as requisições, após finalizar a requisição, os processos entram novamente em espera até que seja delegado outra tarefa para o mesmo.

  Todavia, o processo *dispatcher* pode ser o gargalo do servidor, além de um ponto único de falha do mesmo, pois todas as requisições são delegadas por ele. Ainda, caso a carga

no servidor seja muito alta e o número de processos no pool não tenha sido definido de maneira adequada, o servidor pode ser forçado a criar novos processos.

- ***Thread* por Requisição:** Esse processo é similar "Processo por Requisição", entretanto são usados *threads* no lugar de processos. Para cada requisição é criado uma *thread*.

  O custo de se criar uma nova *thread* é menor que se criar um processo. As *threads* consomem menos recursos da máquina que um processo.

  Algumas vantagens são: As *threads* podem compartilhar o mesmo endereçamento. Já no processo, é necessário copiar o espaço de endereçamento para os processos filhos. Outra vantagem é que enquanto uma *thread* espera por E/S (como aguardar pela leitura dos dados no disco ou aguardar pela recepção e envio dos pacotes pela rede), o controle é passado para outra *thread*, responsável por uma requisição diferente, o que acaba resultando numa utilização bem mais eficiente dos recursos da máquina.

- ***Pool de Threads*:**

  Por fim, a arquitetura é similar a "Pool de Processos"a porem se utiliza *threads*, que são criadas quando o servidor *Web* é iniciado, com o objetivo de eliminar a sobrecarga de criação de uma nova para cada requisição que chega ao servidor.

3 Trabalhos Relacionados

Este capitulo apresenta alguns trabalhos recentes e dados utilizados em gerenciamento, monitoração e analise de desempenho de redes de computadores. Os trabalhos foram organizados por assunto.

3.1 Avaliação de desempenho em fenômenos de rajada

No artigo de Centurion et al. (2012), é apresentado uma avaliação de desempenho voltada para serviços *Web*, onde o autor cria um ambiente para realizar os testes, considerando diferentes características das cargas de trabalho. Dentre as características a serem avaliadas destaca-se a existência ou não dos fenômenos de rajadas. É explicado que este fenômeno é comum no ambiente *Web*, seja em aplicações ou para servidores, e que é de difícil previsibilidade e podem causar gargalos nos serviços disponibilizados na aplicação. Como resultado do artigo, as cargas de trabalho, que tiveram como característica as rajadas foram mais prejudiciais ao desempenho, para todas as variáveis de resposta analisadas.

Já em outro cenário, a tese de Centurion (2015) mostra a caracterização de desempenho dos serviços executados em um ambiente em nuvem simulado e integrada à arquitetura CloudSim-BEQoS, que é uma API do simulador CloudSim.

No trabalho, são consideradas cargas de trabalho que tem como característica as rajadas de diferentes origens, intensidades e variabilidades. Os resultados mostram que a presença de rajadas no processo de chegada das requisições e/ou nas demandas de serviço, ocasiona uma considerável degradação no desempenho e, por isso, devem ser consideradas nos modelos de cargas de trabalhos e nas atividades voltadas para avaliação de desempenho em computação em nuvem. Além é proposto e validado uma metodologia que permite monitorar e determinar a ocorrência de rajadas para a arquitetura (CENTURION, 2015).

O artigo de Gilly et al. (2009), apresenta um estudo que se concentrar no mecanismo de processamento de taxa de chegada, como parte do projeto de um algoritmo *Web* de balanceamento de carga adaptativo. Segundo o autor, a taxa de chegada é uma das métricas mais importantes a serem monitorados em um site da *Web* para evitar o possível congestionamento de servidores *Web*. É definido seis fatores de rajada que devem ser incluídos individualmente em um algoritmo de balanceamento de carga adaptável, que também precisa monitorar continuamente os parâmetros do servidor *Web*, como a taxa de chegada ao servidor ou a utilização da CPU para evitar uma situação inesperada de sobrecarga.

3.2 Método para Avaliação de desempenho de Servidores

A dissertação de Côrte (2002) apresenta um método e métricas usadas para a análise de desempenho de um servidor Web, o qual permite avaliar a disponibilidade, confiabilidade,

facilidades de administração e os recursos. As características básicas e funcionamento do serviço, bem como algumas ferramentas de avaliação de desempenho são descritas.

Ao final foi aplicado o método e métricas a Procempa – Companhia de Processamento de Dados do Município de Porto Alegre, onde se cerificou na prática sua eficácia. Além disto, dados importantes sobre a infra-estrutura *Web* da Procempa foram fornecidos, os quais permitem uma análise do ambiente *Web* atual e futuro da empresa Côrte (2002).

No artigo de (Xianghua Xu et al., 2013), é apresentado um modelo de avaliação chamado pelo autor de: Model Based on the Response Time (Modelo de avaliação de desempenho do servidor da *Web* com base no tempo de resposta ou abreviado MBRT). O mesmo avalia o pico de carga de um servidor da *Web* com determinada configuração. Ele se baseia no relacionamento entre o tempo de resposta e a taxa de transferência quando a taxa de solicitações é menor que o pico de carga. Comparado a outros modelos, o MBRT é simples e eficaz e foi validado do ponto de vista teórico e empírico em um ambiente real .

3.3 Trabalhos relacionados a Gerenciamento de Redes e monitoração

A dissertação de Black (2008) consiste na apresentação e comparação de nove dessas ferramentas de gerenciamento e monitoramento de rede: o CACTI, um front-end para ferramentas de gerenciamento baseado em RRD, ZENOSS¸ ManageOP Engine, BigBrother4, SpiceWorks, Look@LAN, Zabbix e o Nagios.

O objetivo do trabalho foi avaliar os software dentre os analisados e auxiliar o pesquisador a tomar a melhor escolha de acordo com suas necessidades. São observados os parâmetros mais relevantes dentre os procurados pelos administradores de rede, que são: performance, facilidade de utilização e necessidade de recursos tanto de hardware quanto humanos (BLACK, 2008).

Já no artigo de Datt, Goel e Gupta (2015), é apresentado uma lista de atributos necessários para monitorar a infraestrutura e os processos associados ao OpenStack Nova, que é uma plataforma em nuvem de código aberto. Diferentes categorias no monitoramento da infraestrutura são identificadas e os parâmetros de monitoramento associados a eles são listados. A lista proposta foi aplicada a três softwares existentes: Zabbix, Zenoss Zenpack e CollectD, que comumente são usados para monitorar o OpenStack Nova. A lista facilita os administradores de sistema durante a seleção do software de monitoramento existente para o Nova e ajudará os desenvolvedores a selecionar a funcionalidade de monitoramento para o *software* de monitoração.

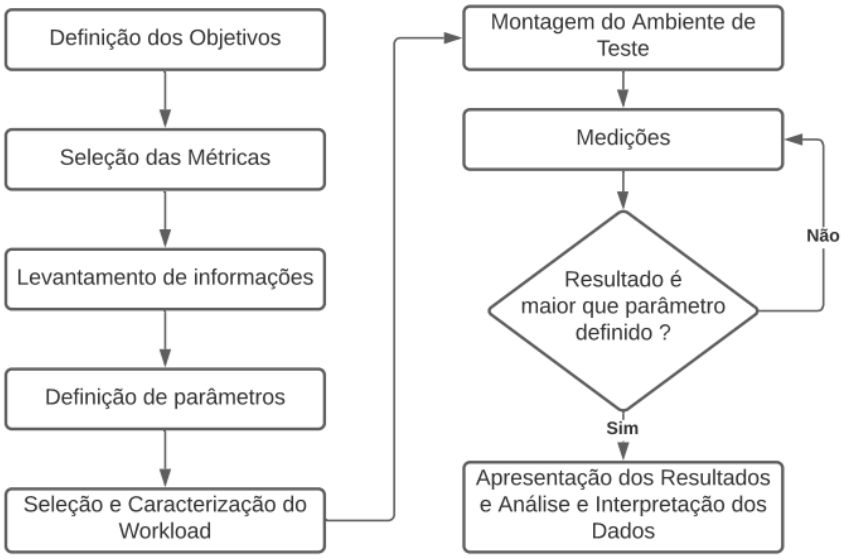
4 Materiais e Métodos

Este capítulo descreve os procedimentos metodológicos e os instrumentos utilizados na realização deste trabalho. Segundo Wazlawick (2017), o presente trabalho tem caráter exploratório, por ser um estudo de caso onde é aplicado o método para a avaliação de servidores *WWW* definido por Côrte (2002) em sua dissertação, com o objetivo de simular períodos de rajada de requisições HTTP e avaliar elementos de hardware sensíveis a este fenômeno e o impacto ao usuário final. Os resultados encontrados foram avaliados juntamente com as métricas definidas na literatura. Para realizar o levantamento bibliográfico referente a pesquisa foram utilizados alguns meios de pesquisa como CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), IEEE Xplore Digital Library, Google Scholar e outras bibliotecas acadêmicas.

4.1 Método

Esta seção descreve o método utilizado, que foi baseado no trabalho de Côrte (2002). Segundo o autor, o método foi fundamentado no livro de Jain (1991) em que é proposto um método genérico de avaliação de desempenho, podendo ser aplicado em diferentes sistemas. Baseando-se em tal, o autor desenvolveu o método composto por 7 etapas com objetivo de avaliar servidores Web. Esse será o método utilizado e descrito neste trabalho, porém com alguns ajustes para se adequar ao objetivo proposto. A Figura 6 ilustra o fluxograma do método:

Figura 6 – Fluxograma do método



Fonte: Elaborada pelo autor

4.1.1 Definição dos Objetivos

Esta etapa define o objetivo do estudo e os limites do sistema a ser analisado, ou seja, compreender o contexto do projeto e onde se quer chegar com a avaliação do sistema. O teste de desempenho se concentra apenas nos itens que são essenciais ao caráter da pesquisa. Isso ajuda a identificar as características de desempenho de maior prioridade nas quais se deve focar os testes.

4.1.2 Seleção das Métricas

As Métricas são medidas ou critérios usados para avaliar o desempenho. Após a etapa de definição do objetivo, é estabelecido as métricas que irão ser utilizadas para avaliar o sistema em seus diferentes aspectos.

A partir das métricas é possível determinar ou validar as características de velocidade, escalabilidade e/ou estabilidade do sistema ou aplicativo em teste. Como exemplo, neste trabalho foi escolhido avaliar elementos de *hardware* e o impacto causado ao usuário, a principal métrica utilizada para avaliar sistemas do ponto de vista do usuário é o tempo de respostas (JUNIOR, 2008).

Para avaliar os recursos de *hardware*, pode-se utilizar a taxa de utilização e vazão. Assim é possível avaliar quais os recursos estão sendo usados com maior frequência e definir o número máximo de requisições que podem ser atendidas pelo sistema.

4.1.3 Levantamento de informações

A principal alteração feita no método de Côrte (2002) foi na etapa de levantamentos de informação, permitindo ser aplicado não só em sistema em produção, mas também em sistemas novos, ou seja, que não foram aplicados na linha de produção, como é o caso deste trabalho. Essas informações são usadas como auxílio tanto na definição de parâmetro quanto na seleção da carga de trabalho (*Workload*).

A coleta de dados do sistemas em produção ajuda a definir se o projeto está adequado ao cenário real com mais exatidão. Neste sentido, o levantamento de informações se faz importante, visto que o conhecimento sobre a intenção do sistema, a arquitetura real de *hardware* e *software* implementada e as características do usuário final são ponto importantes para se definir quais os níveis de carga de trabalho devem ser suportadas pelo sistema e os caminhos percorridos pelo usuário para realizar determinadas atividades.

Entretanto, em sistemas novos, ou seja, que não estão em produção, é importante entender como o sistema funciona e suas principais funções. A partir disso, determinar os principais cenários que o usuário pode percorrer. Essas informações podem ser obtidas em pesquisas de mercado, dados históricos, tendências de mercado e assim por diante.

4.1.4 Definição de parâmetros

Após o levantamento de informações, define-se os parâmetros de aceitação do sistema. A definição de parâmetro é o mesmo que identificar os critérios de sucesso do sistemas,

ou seja, por meio dos limites atribuídos nas métricas é possível definir se o projeto está adequado ao cenário real e encontrar problemas de desempenho. Nesta etapa, pode-se estipular algumas métricas durante o ciclo de vida do sistema ou basear-se em pesquisas feitas muitas vezes relacionada IHC (Interação Homem computador) ou se basear no sistema já em produção.

4.1.5 Seleção e Caracterização do *Workload*

Carga de trabalho ou *workload*, é definido como um estímulo aplicado a um componente ou sistema para simular um fenômeno. Para a modelagem de desempenho o *workload* pode-se associar ao número total de usuários, usuários ativos simultâneos, volumes de dados de entrada/saída e volumes de transações em um determinado sistema.

Nesta etapa são definidos os caminhos de usuário previstos que geralmente são tomados para a executar as atividades do aplicativo. Os principais cenários normalmente escolhidos são aqueles que desempenham um papel importante, como por exemplo: as atividades consideradas de risco elevado, as atividades que são mais comumente usadas ou as com um impacto significativo no desempenho.

Desta forma, neste trabalho, a etapa de seleção e caracterização de *workload* consiste em definir a quantidade requisições de serviço ao sistema e quais os caminhos percorridos pelo usuário durante a navegação, como exemplo: fazer *login*, fazer um pedido, navegar no catálogo, realizar cadastro, pesquisar um produto e fazer *logout* entre outros.

4.1.6 Montagem do Ambiente de Teste e Medições

Nesta etapa é desenvolvido o ambiente de teste, isto é, estabelecer o ambiente a ser testado e preparar as ferramentas de geração de carga e de monitoração. Após a finalização da configuração dos mesmos, executar a ferramenta que simula o fenômeno, realizar a coleta dos dados até atingir o limite para os recursos estabelecidos na etapa de definição de parâmetros.

4.1.7 Apresentação dos Resultados e Análise e Interpretação dos Dados

Como próximo e último passo, apresentar os dados referente aos resultados. É de suma importância que a representação seja clara e objetiva, fazendo-se necessário a utilização de gráficos e tabelas. Por fim os resultados são coletados e associados de forma a possibilitar a análise dos mesmos.

4.2 Materiais

Este capítulo faz uma breve introdução aos principais *softwares* e *hardwares* utilizados durante o projeto.

4.2.1 *Softwares*

4.2.1.1 Zabbix

O Zabbix é uma ferramenta de monitoração *open source*, que possibilita monitorar vários dispositivos de rede de computadores, o que ajuda a garantir a integridade do sistema. O mesmo possui funcionalidades de notificação que permitem configurar alertas baseado em praticamente qualquer evento, tais recursos proporcionam uma reação rápida diante de problemas na rede.

Ademais, apresenta ferramentas de customização de relatórios e a visualização de dados armazenados por meio de *dashboards* (painéis com gráficos e relatórios). Algumas outras vantagem da ferramenta são (ZABBIX, c2001-2020):

- Baixa curva de aprendizado.
- Retorno do investimento elevado visto que *downtimes* (tempo de inatividade) são muito caros.
- Instalação e configuração simples.
- Sistema de monitoramento pode ser arquitetado de forma centralizada ou distribuída.
- Dados são armazenados em banco de dados relacional.
- Suporte para SNMP.
- Capacidades de customizar a visualização dos dados.

4.2.1.2 JMeter

JMeter é um software gratuito, *open source*, multiplataforma e desenvolvido em Java, sendo projetado para testar e medir o desempenho de aplicações. Ele pode ser usado para simular uma carga pesada em um servidor, grupo de servidores, rede ou objeto, para testar sua força ou para analisar o desempenho geral sob diferentes tipos de carga. Além, a ferramenta de teste de desempenho permite avaliar tipos diferentes de aplicativos, servidores e protocolos, como (JMETER, 2014):

- Web - HTTP, HTTPS (Java, NodeJS, PHP, ASP.NET,. . .)
- Serviços da Web SOAP / REST
- FTP
- LDAP
- Middleware orientado a mensagens (MOM) via JMS
- Correio - SMTP (S), POP3 (S) e IMAP (S)
- Comandos nativos ou scripts de shell

- TCP
- Objetos Java

Para criar o *script* que simula o usuário, a ferramenta possui um "Gravador de *Script* de Teste HTTP", que ajuda o testador a registrar os passos do usuário e executar suas atividades em relação ao alvo de teste.

Basicamente, durante a gravação do *script*, o JMeter cria um servidor *proxy*, que atua como intermediário entre o testador e a aplicação, fazendo com que todas as requisições passem por ele. O servidor *proxy* possibilita que o JMeter assista e registre a atividade do testador enquanto estiver navegando no aplicativo da Web com um navegador, facilitando a criação de testes automatizados.

4.2.1.3 VirtualBox

O VirtualBox é um *software open source*, gratuito, que permite criar, gerenciar e executar máquinas virtuais (VMs), que são computadores cujos componentes de *hardware* são emulados pela máquina do usuário, ou seja, o computador que executa o programa. O mesmo pode ser executado nas plataformas *Windows*, Mac OS X, Linux e Solaris (VIRTUALBOX, s.d.).

4.2.1.4 MySQL Workbench

O MySQL Workbench é uma ferramenta de banco de dados que permite o desenvolvimento, administração e manutenção de SQL em um único ambiente de desenvolvimento integrado para o sistema de banco de dados MySQL (MYSQL, c2020).

4.2.1.5 Internet Information Services

O IIS (*Internet Information Services*) é um servidor *Web* criado pela *Microsoft* para seus sistemas operacionais. O mesmo permite hospedar sites na internet por meio do protocolo HTTP. Ademais, fornece um conjunto abrangente de ferramentas para ajudar o gestor a administrar e configurar servidores *Web*, sites e aplicativos (TEMPLIN, 2007).

4.2.2 *Hardware*

Para o desenvolvimento deste trabalho foi utilizados duas máquinas físicas e um roteador/hub que permite a conexão das mesmas por meio de uma rede LAN (*Local Area Networks*, ou Redes Locais), o qual permite interligar computadores presentes dentro de um mesmo espaço físico. Abaixo as características das máquinas e o modelo do roteador:

- Maquina A: Processador Intel(R) Core(TM) i5-4670 CPU @ 3.40GHz 3.40GHz, memória RAM 8,00 GB DDR3.
- Máquina B: Processador Intel(R) Core(TM) i7-3632QM CPU @ 2.20GHz 2.20GHz, memória RAM 8,00 GB DDR3.
- Roteador Askey rtf8115vw.

5 Desenvolvimento

Como foi proposto, o desenvolvimento será dividido igualmente à metodologia, relatando o que foi feito em cada etapa do trabalho.

5.1 Definição dos Objetivos

Aplicar um procedimento teste de performance para determinar ou validar as características de velocidade, escalabilidade e estabilidade do sistemas *Web* quando atingido por uma rajada de usuários. O desempenho está relacionado à obtenção de tempos de resposta e níveis de utilização de recursos de *hardware* que atendem aos objetivos de desempenho do projeto ou produto.

5.2 Seleção das Métricas

Na monitoração do *hardware* foi utilizado a ferramenta Zabbix, o qual traz diversas possibilidades de itens monitoráveis, esses que podem ser de diversos tipos, como por exemplo: Zabbix agent, Simple checks, SNMP, Zabbix internal, IPMI, JMX monitoring etc. Os quais muitas vezes utilizam de estruturas e meios diferentes para a troca de informações entre o servidor que monitora e o sistema a ser monitorado, e são mais usuais em determinados dispositivos.

Este trabalho utilizou o Agente Zabbix para a coleta das métricas, que possui uma estrutura similar ao SNMP. Para isso, o mesmo é instalado nos *hosts* e permite coletar métricas comuns, específicas de *hardware* e sistema operacional, como CPU, memória RAM, memória em disco, interface de rede e etc.

Além disso, o agente Zabbix possibilita a coleta de itens de forma personalizada, por meio de *scripts* ou programas externos é possível customizar as métricas mais complexas e definir ações a serem tomadas diante de determinados eventos.

Na Tabela 2 estão as métricas utilizadas neste trabalho, e as respectivas chaves utilizadas pelo Zabbix que foram baseadas nas métricas propostas no trabalho de Côrte (2002) e no artigo DocsMicrosoft (2017):

Tabela 2 – Métricas utilizadas no Zabbix.

Métrica	Explicação	Chave Zabbix
CPU		
Percentual de utilização da CPU Total	A porcentagem da capacidade de processamento sendo usada.	system.cpu.util
Percentual de utilização da CPU por núcleo	A porcentagem da capacidade de processamento de cada núcleo que está sendo usada pelos processos em execução.	system.cpu.util[0] system.cpu.util[1] system.cpu.util[2] system.cpu.util[3]
Comprimento da fila do processador	É o número de threads em espera que estão prontas para serem executados.	perf_counter_en["\System\Processor Queue Length"]
Interface de rede		
Bits enviados	Tráfego de saída.	net.if.out ["# IFNAME"]
Bits recebidos	Tráfego de entrada.	net.if.in ["# IFNAME"]
Pacotes de saída com erros	Número saída de pacotes de com erros na interface de rede.	net.if.out["Intel(R) PRO/1000 MT Desktop Adapter",errors]
Pacotes de entrada com erros	Número de entrada pacotes com erros na interface de rede.	net.if.in["Intel(R) PRO/1000 MT Desktop Adapter",errors]
Pacotes de saída descartados	Número de pacotes de saída descartados na interface de rede.	net.if.out["Intel(R) PRO/1000 MT Desktop Adapter",dropped]
Pacotes de entrada descartados	Número de pacotes de entrada descartados na interface de rede.	net.if.in["Intel(R) PRO/1000 MT Desktop Adapter",dropped]
Disco Rígido		
Média de tempo por transferência	Tempo médio de leitura e gravação de dados no disco.	perf_counter[\ PhysicalDisk (0 C:)\Avg. Disk sec/Transfer]
Fila de disco média atual	É o número de solicitações pendentes no disco.	vfs.dev.queue_size [CurrentDiskQueue-Length.0 C:]
Espaço em disco usado	Armazenamento usado em bytes.	vfs.fs.size[C:,used]
Espaço em disco total	Espaço total em Bytes.	vfs.fs.size[C:,total]
Memória RAM		
Utilização de memória RAM em bytes	Memória utilizada por todos os processos sendo executados.	vm.memory.size[used]
Memória total em bytes	Total de memória no dispositivo.	vm.memory.size[total]

Fonte: Elaborada pelo autor.

Para avaliar o impacto causado ao usuário, a principal métrica a ser analisada é o tempo de resposta (JUNIOR, 2008). Para isso, foi utilizado o programa JMeter que além de realizar testes automatizados, ele também traz alguns relatórios com as métricas demonstradas na Tabela 3:

Tabela 3 – Métricas JMeter.

Métrica	**Explicação**
Amostra	Número de pedidos enviados.
Média	Média aritmética para todos os tempos resposta.
Min	Tempo de resposta mínimo.
Max	Tempo máximo de resposta.
Desvio Padrão	O desvio padrão mede a distância média dos valores para sua média.
Taxa de erros	Porcentagem de testes com falha.
Vazão	Número de solicitações por segundo enviadas pelo Jmeter.
Média Bytes	Tamanho médio da resposta.

Fonte: Elaborada pelo autor.

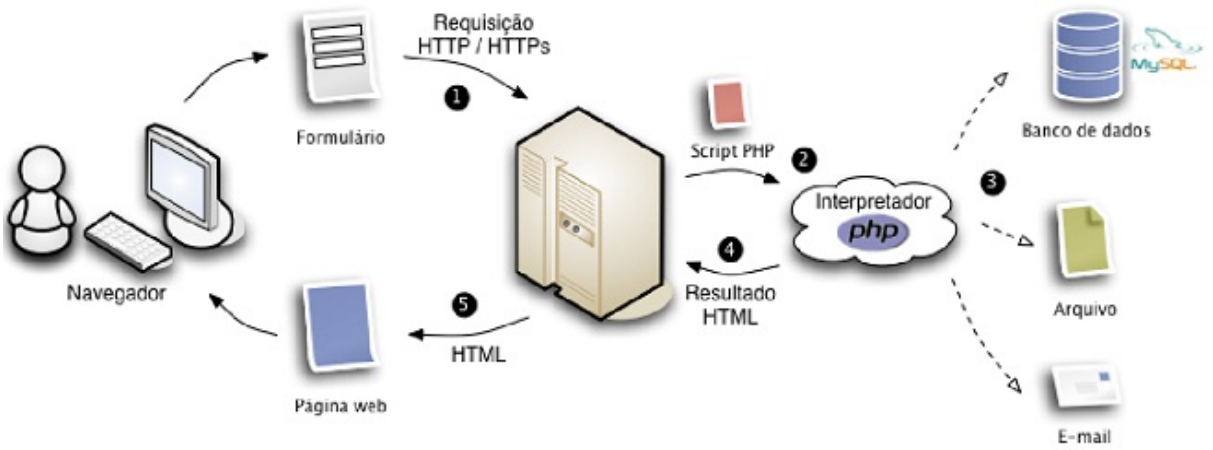
A partir das métricas colhidas é possível ter um panorama de caráter geral, levando em conta os recursos hardware e a qualidade dos serviços disponibilizados pelo sistema, que é medida com base no tempo de resposta.

5.3 Levantamento de informações e Definição de Parâmetros

O sistema *Web* testado possui a combinação de tecnologia PHP (Pré-Processador de Hipertexto) com o MySQL, que é um sistema gerenciador de banco de dados relacional de código aberto, e HTML (*Hypertext Markup Language*) que é uma linguagem de marcação utilizada na construção de páginas na *Web*.

Portanto, o sistema *Web* criado consiste na combinação destas três tecnologias, onde o usuário para acessar o sistema, terá que efetuar um *login*, poderá realizar um cadastro, visualizar um relatório de todos os cadastros criados e efetuar o *logout*. A Figura 7 ilustra funcionamento da tecnologia empregada no sistema *Web*:

Figura 7 – Esquema servidor *Web*



Fonte: (MARINHO, 2014)

Cabe salientar, que o objetivo do trabalho não está no desenvolvimento do sistema *Web*, portanto se criou um cenário simples que possa simular um *site* comum encontrado no mercado.

Na parte de definição de parâmetros, visto que o foco do trabalho é avaliar o impacto ao usuário em momentos de rajada, foi utilizado o tempo de resposta. Segundo Nielsen (2010), normalmente o tempo de resposta deve ser o mais rápido possível quando se trata de navegação na *Web*. Desta forma, pode-se classificar o tempo de resposta em quatro limites:

Em até 0,1 segundo: É o limite para que o usuário sinta que o sistema está reagindo instantaneamente, o que significa que nenhum *feedback* especial é necessário, exceto para exibir o resultado. Basicamente o usuário não percebe o atraso.

De 0,1 até 1,0 segundo: É o limite para o fluxo de pensamento do usuário permanecer ininterrupto, nesse caso o usuário percebe o atraso, porém ainda não é tratado como algo negativo. Até esse tempo nenhum *feedback* especial é necessário, mas o usuário perde a sensação de operar diretamente nos dados. Em determinados contextos é interessante pois o usuário percebe que houve atualização nos dados.

De 1 a 10 segundos: É o limite para manter a atenção do usuário focada. Para essa situação, os usuários desejam executar outras tarefas enquanto aguardam a conclusão do computador, portanto, apresentar um *feedback* durante o atraso é importante. O acúmulo de tais atrasos pode criar uma experiência desagradável para o usuário.

Após 10 segundos: É definido como o pior cenário, os usuários costumam deixar o site em vez de tentar recuperar o ritmo.

Baseado nesses dados, foi definida a seguinte escala de classificação para o tempo de resposta:

- Tempo de resposta entre 0 a 1 segundos: Representa um cenário excelente.

- Tempo de resposta entre 1,0 a 10 segundos: Deve realizar um levantamento da problemática, se possível resolvê-la, se não, oferecer um *feedback* para o usuário.
- Tempo de resposta maior que 10 segundo: É inviável, nesta situação provavelmente o usuário vai desistir de realizar as operações.

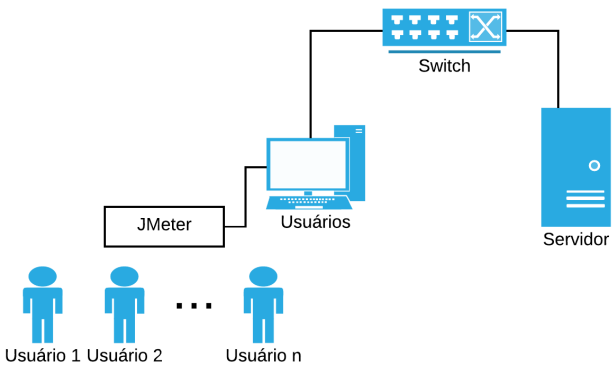
5.4 Seleção e Caracterização do *Workload*

Um teste de desempenho é uma investigação técnica realizada para determinar ou validar as características de capacidade de resposta, velocidade, escalabilidade e/ou estabilidade do produto em teste.

Para este trabalho foi aplicado um grande número de requisições HTTP contendo pesquisas ao banco de dados, com objetivo de estressar ao máximo o servidor e simular algumas características comuns de sites que são encontrados na *Web*.

Desta forma, é necessário criar um teste automatizado, onde será determinado o que os usuários irão realizar durante a navegação, permitindo avaliar o sistema completo ou módulos do sistema. Assim, foi utilizado o Apache JMeter, ferramenta que permite simular requisição de serviços. A Figura 8 representa o diagrama de rede exemplificando o funcionamento do JMeter:

Figura 8 – Fluxograma de rede



Fonte: Elaborada pelo autor

Assim, a ferramenta simula uma determinada quantidade de usuários efetuando os testes automatizados a um servidor, o número de usuário vai depender do poder de processamento da máquina que irá executar o programa. O *Script* criado para esse trabalho foi:

a) Acesso a URL do Sistema;

b) Realizar login;

c) Realizar um cadastro no banco de dados;

d) Visualizar relatório de cadastro;

e) Realizar o logout do site;

Outro ponto importante é que foi inserido temporizadores, que representam os chamados "*think times*", ou tempo entre as requisições, esse que simula o tempo que o usuário permanece em uma página antes de realizar outra requisição. Sem o temporizador as requisições são enviadas entre espaços muito curto de tempo, o que seria inviável para um usuário humano. Então foi inserido um tempo constante entre cada requisição, de 2 segundos, mais um valor aleatório entre 0 a 3 segundo para que cada usuário realizasse tempos distintos e aceitáveis.

5.5 Montagem do Ambiente de Teste e Medições

Para montagem do cenário foi utilizado o VirtualBox, que é um programa de virtualização da Oracle que permite instalar e executar diferentes sistemas operacionais em um único computador. Foram criados duas máquinas virtuais, uma para o servidor Zabbix e a outra para o servidor *Web* estas que foram empregadas em máquinas físicas diferentes, ou seja, em uma das máquinas foi virtualizado o servidor Zabbix e executado o programa JMeter, já na outra foi virtualizado o servidor *Web*.

Na máquina virtual que será avaliada, foi instalado o Windows Server 2019, juntamente com IIS (*Internet Information Services*) que é um servidor *Web* criado pela Microsoft. Também, nesta mesma máquina virtual, foi instalado o servidor de banco de dados MySQL. A Tabela 4 mostra as especificações de hardware da máquina física e da máquina virtual utilizada para o servidor *Web*, servidor Zabbix e JMeter:

Tabela 4 – Características dos equipamentos do ambiente de teste

Função	**Hardware**	**Software**
Servidor Web	Processador Intel(R) Core(TM) i5-4670 CPU @ 3.40GHz 3.40GHz, memória RAM 8,00 GB	Virtualização: Windows Server 2019, 4 processadores, memória RAM 4096
Cliente Web	Processador Intel(R) Core(TM) i7-3632QM CPU @ 2.20GHz 2.20GHz, memória RAM 8,00 GB	Jmeter; Virtualização do servidor Zabbix

Fonte: Elaborada pelo autor.

Como sistema de monitoração, foram avaliados duas ferramentas, o Monitor de Desempenho do *Windows* (PerfMon) e o Zabbix, ferramenta utilizada para monitoração de recursos de rede. O *PerfMon* possui gráficos mais precisos visto que a frequência de coleta é maior que a do Zabbix, porém é uma ferramenta não genérica, podendo ser aplicada apenas nos

sistemas operacionais *Windows*, outro ponto é que não foi avaliado essa ferramenta de forma remota.

Já o Zabbix, como já dito, possui menor frequência de coleta de dados quando comparado ao Perfamon, porém é um sistema genérico podendo ser aplicado em diversos sistemas diferentes, como Linux, Windows e outros dispositivos de rede.

Outro ponto é que o Zabbix tem uma interface *Web* onde é possível montar gráficos, *dashboards*, inserir alertas, dentre outras ferramentas que ajudam o gerente de rede monitorar tais sistemas. Visando essas características, o Zabbix foi utilizado para a coleta dos dados.

Em fim, é importante destacar que os dados coletados só serão válidos para sistemas idênticos, pois características como *hardware*, sistema operacional, sistemas *Web*, dentre outras tecnologias utilizadas influenciam diretamente nos resultados. Porém o método utilizado para a montagem dos testes de desempenho poderão ser aplicados em outros sistemas.

5.6 Apresentação dos Resultados

Nesta etapa é apresentado os dados que foram utilizados durante a fase de análise dos resultados. Os dados relacionados ao tempo de resposta foram retirados da ferramenta JMeter, os relacionados ao desempenho de *hardware* do Zabbix. Para mais detalhes, os gráficos e tabelas colhidos para os testes de 100, 200, 300 e 400 usuários, foram disponibilizados nos Apêndices A, B, C e D, respectivamente.

5.6.1 Teste 1 - 100 Usuários

Tabela 5 – Métricas Jmeter para 100 Usuários

Rótulo	Amostra	Média	Min	Max	Desvio Padrão	% Erros	Vazão
Login	100	0,054 s	0,029 s	0,142	0,021s	0,000%	9,9 r/s
Cadastro	100	0,342 s	0,106 s	0,809	0,162s	0,000%	5,6 r/s
Consulta	100	0,085 s	0,063 s	0,157	0,015s	0,000%	9,3 r/s
Logout	100	0,044 s	0,024 s	0,090	0,014s	0,000%	8,7 r/s
Total	400	0,131 s	0,024 s	0,809	0,147s	0,000%	12,2 r/s

Fonte: Elaborada pelo autor.

Tabela 6 – Métricas CPU para 100 Usuários: Taxa de utilização e fila de threads

Núcleo 0	Núcleo 1	Núcleo 2	Núcleo 3	Núcleos Média Total	Fila Threads
20,3602%	19,2417%	23,492%	27,1648%	22,1188 %	5

Fonte: Elaborada pelo autor.

Tabela 7 – Métricas Disco Rígido para 100 Usuários

Temp. Tranf.	Média Fila	Espaço Utilizado
329 ms/transf.	0	29.63%

Fonte: Elaborada pelo autor.

Tabela 8 – Métricas tráfego na placa de rede para 100 Usuários

Bits Enviados	Bits Recebidos
44 Mbps	89,64 Kbps

Fonte: Elaborada pelo autor.

Tabela 9 – Descartes/Erros na placa de rede para 100 Usuários

Entrada erros	Saída erro	Entrada Desc.	Saída Desc.
0	0	0	0

Fonte: Elaborada pelo autor.

Tabela 10 – Utilização de Memória RAM para 100 Usuários

Memória Utilizada	Total de Memória
1,96 GB	4,00 GB

Fonte: Elaborada pelo autor.

5.6.2 Teste 2 - 200 Usuários

Tabela 11 – Métricas Jmeter para 200 Usuários

Rótulo	Amostra	Média	Min	Max	Desvio Padrão	% Erros	Vazão
Login	200	0,132 s	0,029 s	0,380 s	0,102 s	0,000%	18,6 r/s
Cadastro	200	3,821 s	2,590 s	4,881 s	0,532 s	0,000%	9,5 r/s
Consulta	200	0,669 s	0,084 s	1,763 s	0,450 s	0,000%	16,6,3 r/s
Logout	200	0,837 s	0,023 s	3,945 s	1,124 s	0,000%	15,3 r/s
Total	800	1,365 s	0,023 s	4,881 s	1,587 s	0,000%	21,6 r/s

Fonte: Elaborada pelo autor.

Tabela 12 – Métricas CPU para 200 usuários: Taxa de utilização e fila de threads

Núcleo 0	Núcleo 1	Núcleo 2	Núcleo 3	Núcleos Média Total	Fila Threads
32,067%	32,730%	32,727%	36,588%	33,316 %	30

Fonte: Elaborada pelo autor.

Tabela 13 – Métricas Disco Rígido para 200 Usuários

Temp. Tranf.	Média Fila	Espaço Utilizado
415 ms/transf.	2	29.63%

Fonte: Elaborada pelo autor.

Tabela 14 – Espaço livre em disco 200 Usuários

Bits Enviados	Bits Recebidos
81,35 Mbps	1,7 Mbps

Fonte: Elaborada pelo autor.

Tabela 15 – Descartes/Erros na placa de rede para 200 Usuários

Entrada erros	Saída erro	Entrada Desc.	Saída Desc.
0	0	0	0

Fonte: Elaborada pelo autor.

Tabela 16 – Utilização de Memória RAM para 200 Usuários

Memória Utilizada	Total de Memória
2,06 GB	4,00 GB

Fonte: Elaborada pelo autor.

5.6.3 Teste 3 - 300 Usuários

Tabela 17 – Métricas Jmeter para 300 Usuários

Rótulo	Amostra	Média	Min	Max	Desvio Padrão	% Erros	Vazão
Login	300	1,407 s	0,026 s	2,433 s	0,731 s	0,000%	24,2 r/s
Cadastro	300	3,757 s	2,616 s	5,141 s	0,551 s	0,000%	13,4 r/s
Consulta	300	0,939 s	0,074 s	2,536 s	0,531 s	0,000%	19,3 r/s
Logout	300	1,055 s	0,017 s	4,383 s	1,247 s	0,000%	19,4 r/s
Total	1200	1,790 s	0,017 s	5,141 s	1,410 s	0,000%	30,3 r/s

Fonte: Elaborada pelo autor.

Tabela 18 – Métricas CPU para 300 Usuários: Taxa de utilização e fila de threads

Núcleo 0	Núcleo 1	Núcleo 2	Núcleo 3	Núcleos Média Total	Fila Threads
36,338%	35,293%	38,361%	42,312%	37,806 %	35

Fonte: Elaborada pelo autor.

Tabela 19 – Métricas Disco Rígido para 300 Usuários

Temp. Tranf.	Média Fila	Espaço Utilizado
470 ms/transf.	2	29.32%

Fonte: Elaborada pelo autor.

Tabela 20 – 300 Usuários

Bits Enviados	Bits Recebidos
123,71 Mbps	2,56 Mbps

Fonte: Elaborada pelo autor.

Tabela 21 – Descartes/Erros na placa de rede para 300 Usuários

Entrada erros	Saída erro	Entrada Desc.	Saída Desc.
0	0	0	0

Fonte: Elaborada pelo autor.

Tabela 22 – Utilização de Memória RAM para 300 Usuários

Memória Utilizada	Total de Memória
1,84 GB	4,00 GB

Fonte: Elaborada pelo autor.

5.6.4 Teste 4 - 400 Usuários

Tabela 23 – Métricas Jmeter para 400 Usuários

Rótulo	Amostra	Média	Min	Max	Desvio Padrão	% Erros	Vazão
Login	400	0,663 s	0,029 s	1,557 s	0,408 s	0,000%	34,8 r/s
Cadastro	400	15,614 s	12,342 s	24,143 s	2,858 s	0,000%	9,9 r/s
Consulta	400	7,279 s	0,129 s	13,549 s	3,197 s	0,000%	15,6 r/s
Logout	400	6,172 s	0,150 s	17,094 s	3,672 s	0,000%	15,8 r/s
Total	1600	7,432 s	0,029 s	24,143 s	6,050 s	0,000%	26,5 r/s

Fonte: Elaborada pelo autor.

Tabela 24 – Métricas CPU para 400 Usuários: Taxa de utilização e fila de threads

Núcleo 0	Núcleo 1	Núcleo 2	Núcleo 3	Núcleos Média Total	Fila Threads
56,3799%	53,5396%	56,5598%	60,1440%	56,4814 %	95

Fonte: Elaborada pelo autor.

Tabela 25 – Métricas Disco Rígido para 400 Usuários

Temp. Tranf.	Média Fila	Espaço Utilizado
461 ms/transf.	1	29.63%

Fonte: Elaborada pelo autor.

Tabela 26 – Métricas tráfego na placa de rede para 400 Usuários

Bits Enviados	Bits Recebidos
185,87 Mbps	3,42 Mbps

Tabela 27 – Descartes/Erros na placa de rede para 400 Usuários

Entrada erros	Saída erro	Entrada Desc.	Saída Desc.
0	0	0	0

Tabela 28 – Utilização de Memória RAM para 400 Usuários

Memória Utilizada	Total de Memória
2,26 GB	4,00 GB

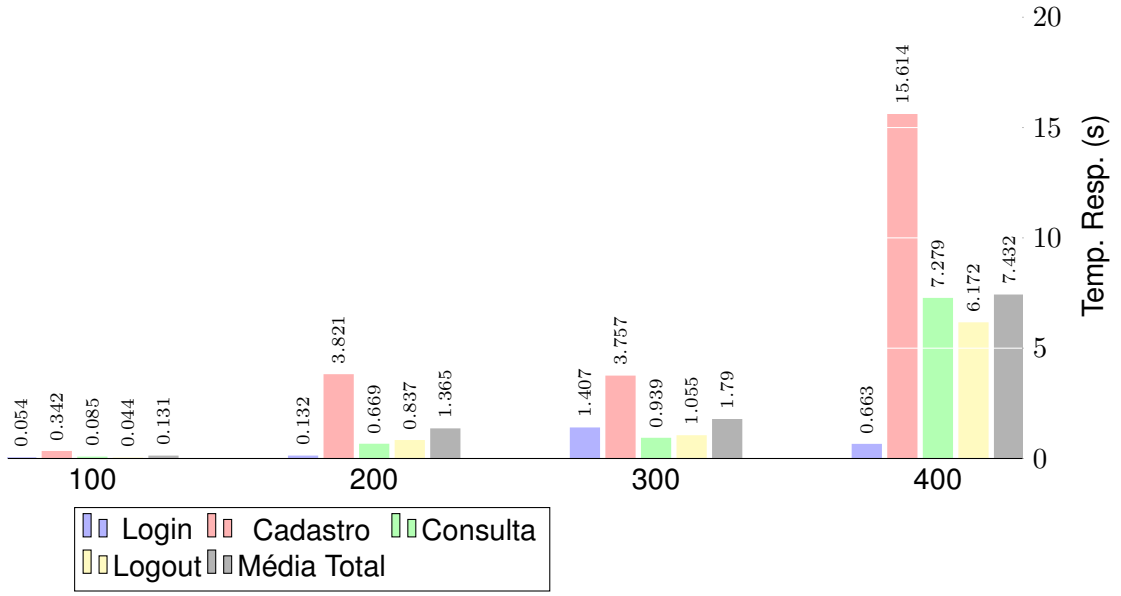
5.7 Análise dos Resultados

Este capítulo apresenta a análise dos resultados encontrados com o experimento realizado no sistema. Os mesmos mostraram alguns fatores que influenciaram no desempenho do servidor *Web*, relacionados a *hardware* e ao desempenho da aplicação.

Dentre as métricas foram utilizados algum limites da ferramenta *Performance Analysis of Logs* (PAL), o qual lê os registros do contador do Monitor de Desempenho do *Windows (PerfMon)* e os analisa com base em limites conhecidos. Os mesmos foram definidos pelas equipes de produtos da *Microsoft*, incluindo o *BizTalk Server* e membros do suporte da *Microsoft*, para auxiliar na análise de elementos de *hardware* (DOCSMICROSOFT, 2017).

Em primeira análise, foi avaliado o tempo de resposta. A Figura 9 representa os tempos de respostas para cada etapa do teste em função número de usuários:

Figura 9 – Gráfico - Tempo de resposta x Usuário



Fonte: Elaborada pelo autor.

Como já esperado o teste com melhor resultado foi o de 100 usuários, com todos os tempos abaixo do definido. Na segunda e terceira interação é possível observar que a etapa de cadastro não obteve êxito, atingindo um tempo de resposta acima de 1 segundo. Todavia, as outras etapas ainda apresentaram bons resultados, ou seja, próximos do definido. Na última interação pode-se observar um grande aumento no tempo de resposta com tempos acima de 10 segundos ou próximos para as etapas de cadastro, consulta e *logout*.

Assim, a etapa de cadastro é a que mais demanda tempo da máquina para ser processada, visto que a partir da segunda interação já ultrapassa o limite definido para um cenário eficiente. Outro ponto observado foi o desvio padrão, que é uma medida que expressa o grau de dispersão de um conjunto de dados. Quanto menor o valor do desvio padrão, mais uniforme serão os tempos de resposta coletados. A Tabela 29 apresenta as médias dos desvios padrões coletados nos testes de 100, 200, 300 e 400 usuários:

Tabela 29 – Desvio padrão do tempo de resposta

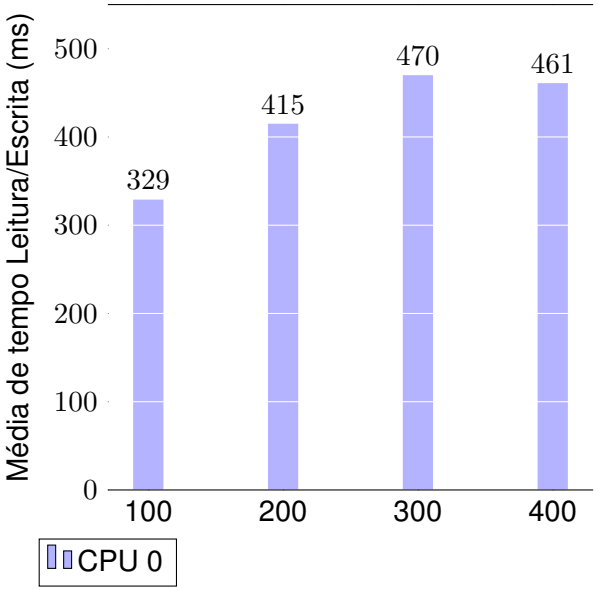
Teste	Desvio padrão
100 Usuários	0,147 s
200 Usuários	1,587 s
300 Usuários	1,410 s
400 Usuários	6,050 s

Fonte: Elaborada pelo autor.

Com base nos dados, é possível observar que a medida em que o número de usuários aumenta o sistema apresenta tempos de respostas mais dispersos em relação a sua média, o que demonstra instabilidade do sistema em realizar as tarefas.

Um dos fatores que podem estar relacionados com os altos tempos de respostas é o tempo de leitura e escrita de disco. A Figura 10 representa o tempo de transferência de disco rígido em função do número de usuários:

Figura 10 – Disco Rígido - Tempo de transferência x Usuário



Fonte: Elaborada pelo autor.

Segundo DocsMicrosoft (2017), a maneira mais confiável detectar um gargalo de desempenho do disco é medindo a latência de leitura e gravação. Caso os tempos de resposta ultrapassem 25 milissegundos, que é um limite conservador, podem ocorrer lentidão perceptível e problemas de desempenho que afetam os usuários.

Baseado nessa informação, todas as médias de tempo de leitura/escrita em disco apresentaram valores acima de 25 milissegundos, entretanto para o teste de 100 usuários o sistema obteve resultados satisfatórios, o mesmo ainda consegue atender essa demanda de forma eficiente, sem que o usuário seja prejudicado com altos tempos de resposta. Entretanto, no sistema apresentado esse seria um recurso que poderia ser aprimorado dependendo do objetivo do sistema.

Ademais foram avaliados a média do comprimento de fila em disco e o espaço utilizado, como mostra a Tabela 30:

Tabela 30 – Métricas do disco rígido

Num. Usuários	Média Fila	Espaço Utilizado
100	0	29.63%
200	2	29.63%
300	2	29.32%
400	1	29.63%

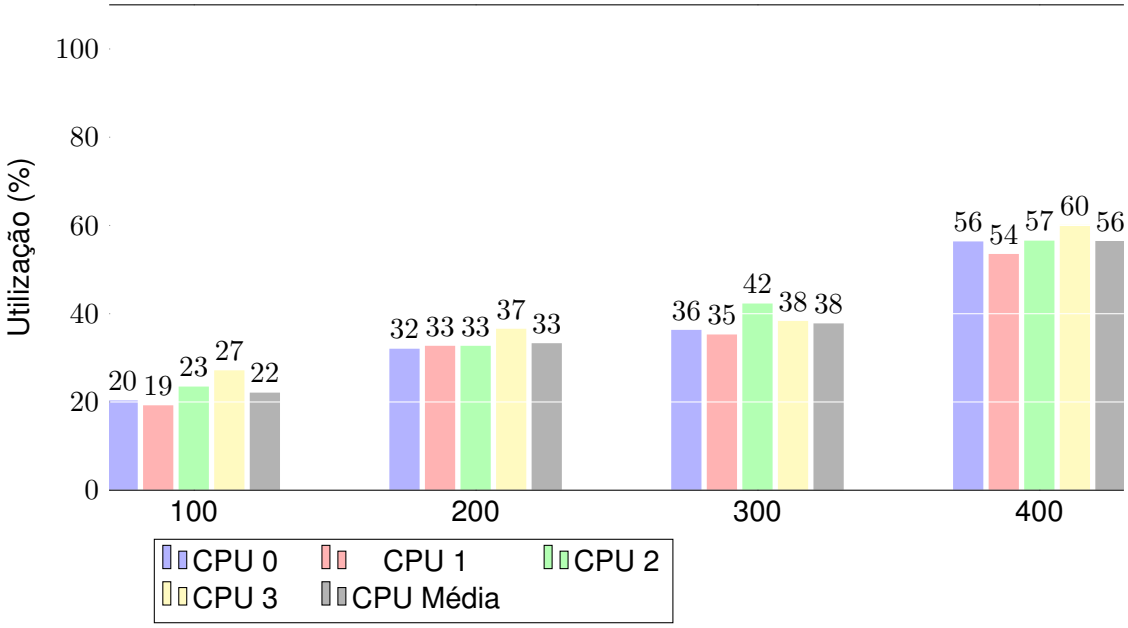
Fonte: Elaborada pelo autor.

Quanto ao espaço livre em disco não foi observado grandes interferências, visto que o desempenho só deve ser afetado quando o espaço disponível na unidade de disco for inferior a 30% (DOCSMICROSOFT, 2017). Isso resulta do espaço livre restante ficar localizado próximo ao eixo do disco, que opera em um nível de desempenho inferior. A falta de espaço livre em disco pode causar grandes impactos ao desempenho.

Já a média do comprimento da fila de disco é o número de solicitações que estão pendentes no disco no momento em que os dados de desempenho são coletados. Valores acima de 0 significam que o disco não é capaz de atender às solicitações de E/S tão rápido quanto elas estão sendo feitas. No sistema, identificou-se a incapacidade de atender todas as requisições gerando fila nos testes de 200, 300 e 400 usuários,

Em relação a CPU, de acordo com DocsMicrosoft (2017), a taxa de utilização do processador é a porcentagem de tempo decorrido que o processador gasta para executar as *threads* não ocioso. É importante verificar se a essa taxa não ultrapassou 60%. A Figura 11 representa a taxa de utilização em todos os núcleos, e a média total em relação ao número de usuários:

Figura 11 – Gráfico - Utilização CPU x Usuário



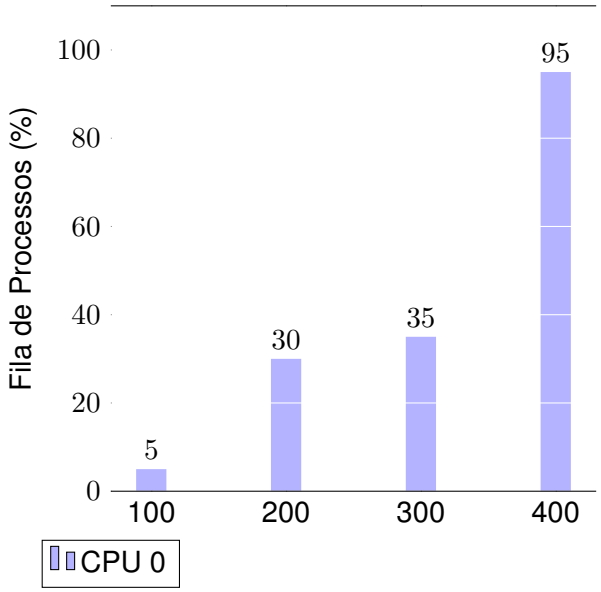
Fonte: Elaborada pelo autor.

Em sistemas com vários núcleos/CPU, é importante examinar tanto o uso combinado da CPU quanto de modo individual, pois existem aplicações que podem assumir o controle total do processador. Nestes sistemas, os núcleos podem apresentar diferentes taxas de utilização, o que seria de difícil visualização no gráfico combinado dos núcleos. Neste trabalho o sistema não possui controle total dos núcleos, logo o sistema operacional faz a distribuição de carga para cada núcleo, evitando a sobrecarga de apenas um.

Houve variações consideradas de utilização à medida que aumenta o número de requisições/usuários, porém a porcentagem de utilização da CPU não ultrapassou 60% em nenhum dos testes o que não remete a um gargalo.

Outra métrica importante é o comprimento da fila do processador que permite saber o número de processos que estão na fila de espera da CPU. Baseado em DocsMicrosoft (2017), é aceitável uma fila com menos de 10 *threads* por processador, sem levar em conta a carga de trabalho. Abaixo a Figura 12 relaciona a fila de processos da CPU com o número de usuários:

Figura 12 – Gráfico - Fila CPU x Usuário



Fonte: Elaborada pelo autor.

Conforme o gráfico, o único teste que ultrapassou os limites foi o teste de 400 usuários, visto que foi alocado para a máquina virtual 4 núcleos o que daria um total de 40 processos. Nessa situação, se faz necessário avaliar juntamente com a utilização da CPU.

No caso em que a fila seja maior que a calculada e o valor de utilização de CPU esteja acima de 90% é um indício de gargalo no processador. Essa situação pode ser causada por dois motivos: a própria aplicação, quando a criação de um grande número de *threads*, o que gera muita troca de contexto e/ou o próprio processador não é capaz de finalizar os processos em tempo adequado.

Porém, no teste de 400 usuários o resultado foi de uma fila grande e processamento abaixo de 60%. Isso significa que o processador não é um gargalo. Motivos comuns para esse resultado é que as solicitações de tempo do processador chegam aleatoriamente e *threads* com tempos irregulares de processamento.

Em relação a placa de rede não foi observado influência significativa no desempenho da aplicação, visto que a velocidade máxima especificada pelo fabricante é de 1000 Mbps e a velocidade máxima atingida durantes os testes 1, 2, 3 e 4 não ultrapassou esses limites, como mostra a Tabela 31:

Tabela 31 – Métricas tráfego na placa de rede

Num. Usuários	Bits Enviados	Bits Recebidos
100	44 Mbps	89,64 Kbps
200	81,35 Mbps	1,7 Mbps
300	123,71 Mbps	2,56 Mbps
400	185,87 Mbps	3,42 Mbps

Fonte: Elaborada pelo autor.

Também foi observado o número de pacotes de entrada/saída descartados e com erro na placa de rede, os dados estão dispostos na Tabela 32:

Tabela 32 – Descartes/Erros na placa de rede

Num. Usuários	Entrada erros	Saída erro	Entrada Desc.	Saída Desc.
100	0	0	0	0
200	0	0	0	0
300	0	0	0	0
400	0	0	0	0

Fonte: Elaborada pelo autor.

Caso os resultados apresentassem descartes/erros de pacotes, o problema poderia tanto estar associado aos recursos de hardware ou software, quanto à interferência de rede. Entretanto, não houve descarte e erros de pacotes na saída e entrada da interface de rede, o que remete a um bom funcionamento. Outro ponto é que se usou uma rede interna para o sistema, logo a interferência gerada pela rede foi minimizada.

Por fim, foi avaliado a memória RAM a partir da quantidade de “Mbytes Ulilizados” , como mostra a Tabela 33:

Tabela 33 – Utilização de Memória RAM

Num. de Usuários	Memória Utilizada	Total de Memória
100	1,96 GB	4,00 GB
200	2,06 GB	4,00 GB
300	1,84 GB	4,00 GB
400	2,26 GB	4,00 GB

Fonte: Elaborada pelo autor.

À medida que o espaço livre se torna escasso, o sistema operacional começa a usar o arquivo de paginação de forma mais intensa. O uso do arquivo de paginação afeta o desempenho geral, pois as operações de disco demoram mais do que as operações de memória RAM. Desta forma, o sistema não apresentou resultados ruins, sendo que a utilização de memória RAM não ultrapassou 2,26 GB para um total de 4 GB.

6 Considerações finais

O presente trabalho apresenta um procedimento operacional de avaliação de desempenho de servidores *Web*, onde foram avaliados aspectos relacionados a *hardware* e desempenho da aplicação em uma perspectiva do usuário, quando aplicado uma rajada de requisições.

O objetivo foi alcançado, visto que foi possível aplicar uma metodologia baseada na literatura científica com algumas alterações para o desenvolvimento da avaliação de desempenho focada nos recursos de *hardware* e no impacto ao usuário em servidores *Web*.

A metodologia serviu de suporte para o estudo de caso, onde se avaliou um cenário *Web*. Com base nos resultados, foi possível identificar gargalos no sistema não só nos recursos da máquina, mas também poderia ser utilizado para o aprimoramento da própria aplicação.

Ademais, foi possível avaliar a aplicabilidade das métricas escolhidas, que se mostraram efetivas para diagnosticar o cenário testado quanto a sua escalabilidade, disponibilidade e desempenho dos recursos.

O presente trabalho também contribuiu na área de monitoração, com aplicação em sistemas de monitoramento de servidores *Web*, visto que se buscou utilizar ferramentas genéricas que possam ser empregadas em diferentes sistemas. Além disso, oferece um nível de detalhamento adequado para a análise das métricas escolhidas, o que ajuda a identificar gargalos no sistema.

Embora este trabalho de pesquisa não alcançou, em ambiente de simulação, todas as possibilidades de utilização dos recursos disponíveis, foi possível demonstrar características do sistema em relação ao seu desempenho e ter uma perspectiva do impacto ao usuário em momentos de rajada.

Outro ponto é que não foi abordado todas as métricas que são tratadas na bibliografia e que estão disponíveis para a coleta, sendo um número extremamente grande. Muitas destas não foram empregadas, porém poderiam ser utilizadas para ter uma maior compreensão do sistema.

Por fim, em relação ao sistema analisado, para o teste de 100 usuários, o sistema obteve tempos de resposta abaixo de 1 segundo para todas as etapas, o qual remete ao melhor cenário dentre os apresentados.

Para os testes de 200 e 300 usuários o sistema obteve um resultado intermediário, onde o tempo de resposta de login foi adequado, para as etapas consulta, logout o tempo ficou próximo de 1 segundo, já a etapa de cadastro ultrapassou esse tempo.

Por último, o teste de 400 se mostrou o pior cenário, com alto tempo de resposta nas etapas de cadastro, consulta e logout. Com base nas análises, o resultado se deve ao disco rígido, devido à sua latência de leitura/escrita.

6.1 Trabalhos futuros

Como trabalhos futuros envolvem explorar a possibilidade da adoção/criação de métodos e procedimentos operacionais:

Teste de carga: É realizado para verificar se seu aplicativo pode atender aos objetivos de desempenho desejados.

Teste de stress: Procura determinar ou validar o comportamento de um aplicativo quando ele é enviado para além das condições normais ou de pico de carga.

Teste de capacidade: Determina quantos usuários e/ou transações um determinado sistema suportará e ainda atenderá às metas de desempenho.

Outra proposta de trabalho futuro é desenvolver procedimentos operacionais e métodos na área de testes automatizados, os quais são *scripts* que procuram simular eventos que podem ser estressantes para o sistema. Tais procedimentos tem como objetivo testar a disponibilidade e qualidade dos serviços oferecidos quando atingidos por tais fenômenos.

Por fim, realizar estudos de casos aplicando a metodologia, com objetivo de avaliar sistemas computacionais, buscando aprimorar a avaliação de desempenho não só adicionando novas métricas, mas também avaliando sua aplicabilidade em outros cenários.

Referências

ARANTES, J. A. et al. *Modelo analítico para avaliar plataformas cliente/servidor e agentes móveis aplicado à gerência de redes*. 2012. Dissertação (Mestrado), 2012. Citado nas páginas 22, 23 e 24.

BENICIO, W. E. P. Monitoramento e gerenciamento de redes utilizando zabbix. *Trabalho apresentado ao Curso de Análise e Desenvolvimento de Sistemas do Instituto Federal como requisito para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas*, 2015. Citado na página 16.

BLACK, T. L. Comparação de ferramentas de gerenciamento de redes. 2008. Citado na página 33.

CASALE, G. et al. BURN: Enabling Workload Burstiness in Customized Service Benchmarks. *IEEE Transactions on Software Engineering*, v. 38, n. 4, p. 778–793, jul 2012. ISSN 0098-5589. Disponível em: <http://ieeexplore.ieee.org/document/5928353/>. Citado na página 16.

CASE, J. D. et al. *Simple network management protocol (SNMP)*. [S.l.], 1990. Citado na página 24.

CENTURION, A. M. *Impacto das rajadas no desempenho de serviços executados em ambientes em nuvens*. 2015. 175 p. Tese (Doutorado), 2015. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-10092015-144229/pt-br.php>. Citado nas páginas 15 e 32.

CENTURION, A. M. et al. Impacto da carga de trabalho com rajadas no desempenho de serviços web. *XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. Ouro Preto-MG. SBRC*, p. 422–435, 2012. Citado nas páginas 16 e 32.

CONTESSA, D. F.; POLINA, E. R. *Gerenciamento de Equipamentos Usando o Protocolo SNMP*. [S.l.], 2010. Citado nas páginas 22, 23 e 24.

CÔRTE, L. Método para a avaliação de servidores www no ambiente corporativo. 2002. Citado nas páginas 32, 33, 34, 35 e 39.

COSTA, H. L. A. et al. *Alta disponibilidade e balanceamento de carga para melhoria de sistemas computacionais críticos usando software livre: Um estudo de caso*. 2009. Dissertação (Mestrado) — Universidade Federal de Viçosa, 2009. Citado na página 15.

DATT, A.; GOEL, A.; GUPTA, S. Analysis of Infrastructure Monitoring Requirements for OpenStack Nova. *Procedia Computer Science*, v. 54, p. 127–136, 2015. ISSN 18770509. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1877050915013393>. Citado nas páginas 15 e 33.

DIAS, B. Z.; JR, N. A. Protocolo de gerenciamento snmp. *artigo extraído da Internet*, 2002. Citado nas páginas 22 e 23.

DOCSMICROSOFT. *Using the Performance Analysis of Logs (PAL) Tool - BizTalk Server | Microsoft Docs*. 2017. Disponível em: <https://docs.microsoft.com/en-us/biztalk/technical-guides/using-the-performance-analysis-of-logs-pal-tool>. Citado nas páginas 39, 50, 51, 52 e 53.

FOROUZAN, B. A. *Comunicação de dados e redes de computadores*. Quarta. São Paulo: [s.n.], 2008. ISBN 9788563308474. Citado nas páginas 27, 28, 29 e 30.

FOROUZAN, B. A.; MOSHARRAF, F. *Redes de computadores: uma abordagem top-down*. [S.l.]: AMGH Editora, 2013. Citado na página 18.

GILLY, K. et al. Analysis of burstiness monitoring and detection in an adaptive Web system. *Computer Networks*, v. 53, n. 5, p. 668–679, apr 2009. ISSN 13891286. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1389128608003770>. Citado na página 32.

JAIN, R. K. The art of computer systems performance analysis: techniques for experimental design, measurement, simulation and modeling. Wiley, 1991. Citado na página 34.

JMETER, A. *Apache JMeter - Apache JMeter™*. [S.l.], 2014. Disponível em: <http://jmeter.apache.org/>. Citado na página 37.

JOHNSON, T.; MARGALHO, M. *Avaliação de Desempenho de Sistemas*. 2014. 1–200 p. Citado nas páginas 25, 26 e 27.

JUNIOR, G. G. *Modelo de Servidor Web com Quatro Módulos de Atendimento de Requisições (SWMAR)*. 2008. 79 p. Dissertação (Mestrado) — Instituto de Ciências Matemáticas e de Computação - ICMC-USP, 2008. Citado nas páginas 28, 30 e 35.

KUROSE, J. F.; ROSS, K. W. *Computer networking: a top-down approach*. [S.l.]: Addison-Wesley Reading, 2010. Citado nas páginas 18, 19, 20, 21, 22, 27 e 29.

LEME, A. S. P. *ANÁLISE DE SOLUÇÕES ABERTAS DE GERENCIAMENTO DE REDES EM RELAÇÃO AO PADRÃO FCAPS (ITU-T M.3400) DE GERENCIAMENTO DE REDES*. 2016. Dissertação (Mestrado) — FUNDAÇÃO MINEIRA DE EDUCAÇÃO E CULTURA FACULDADE DE CIÊNCIAS EMPRESARIAIS MESTRADO EM SISTEMAS DE INFORMAÇÃO, 2016. Citado nas páginas 19, 20 e 21.

LINHARES, E. S. et al. Comparativos Entre Os Servidores Web: Apache E Iis. *Revista Computação Aplicada - UNG-Ser*, v. 3, n. 1, p. 35–39, 2014. ISSN 2316-7394. Disponível em: <http://revistas.ung.br/index.php/computacaoaplicada/article/view/1950>. Citado na página 15.

MACEDO, R. T. et al. Redes de computadores. Brasil, 2018. Citado na página 18.

MARINHO, F. *Introdução ao desenvolvimento web com PHP – Aula 1 – Preparando o ambiente para iniciar a programação*. 2014. Disponível em: <https://www.fredericomarinho.com/>. Citado na página 42.

MAURO, D. R.; SCHMIDT, K. J. Snmp essencial. *Rio de Janeiro: Campus*, 2001. Citado na página 24.

MESSIAS, V. R. *Servidor web distribuído com diferenciação de serviços – implementação e avaliação de um protótipo*. 2007. 125 p. Dissertação (Mestrado) — Instituto de Ciências Matemáticas e de Computação - ICMC-USP, 2007. Citado na página 30.

MICHELON, G. A.; HILD, G. F. Reutilização de computadores obsoletos com a implementação de um servidor de terminais gnu/linux. 2009. Citado na página 15.

Miniwatts Marketing Group. *Internet world stats*. 2019. 3 p. Disponível em: <www.internetworldstats.com/stats.htm>. Citado na página 15.

Moraes Junior, H. P. *Ferramentas de avaliação de desempenho para servidores web: análise, implementação de melhorias e testes*. jan 2016. Dissertação (Mestrado) — Universidade de São Paulo, São Carlos, jan 2016. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-29012016-142348/>. Citado nas páginas 25, 26 e 27.

MYSQL. *MySQL Workbench*. c2020. Disponível em: <https://www.mysql.com/products/workbench/>. Citado na página 38.

NIELSEN, J. *Website Response Times*. 2010. 1 p. Disponível em: <https://www.nngroup.com/articles/website-response-times/>. Citado na página 42.

PINHEIRO, J. M. dos S. *Gerenciamento de redes de computadores*. [S.l.]: Agosto, 2002. Citado na página 18.

REHMAN, Z. ur et al. User-side QoS forecasting and management of cloud services. *World Wide Web*, v. 18, n. 6, p. 1677–1716, nov 2015. ISSN 1386-145X. Disponível em: <http://link.springer.com/10.1007/s11280-014-0319-8>. Citado na página 15.

SANTOS, M. T. et al. *Gerência de Redes de Computadores*. Escola sup. Rio de Janeiro: [s.n.], 2015. 304 p. Citado nas páginas 19, 20 e 21.

SILVA, A. C.; OLIVEIRA, S. A. Gerenciamento Descentralizado de Rede com Software Livre. *Anais do VII Congresso Sul Brasileiro de Computação*, p. 1691–1701, 2014. Disponível em: <http://periodicos.unesc.net/sulcomp/article/view/1789>. Citado na página 19.

SILVA, M. J. S. D. Uma abordagem para avaliação de desempenho de serviços web. *Dissertação apresentada ao Programa de Pós-Graduação em Informática Aplicada como exigência parcial à obtenção do título de Mestre.*, 2015. Citado na página 17.

TANENBAUM, A. S. et al. *Redes de Computadores*. Quinta. [S.l.: s.n.], 2011. v. 5. 600 p. Citado nas páginas 15, 29 e 30.

TEMPLIN, R. *Introduction to IIS Architectures*. 2007. Disponível em: <https://docs.microsoft.com/pt-br/iis/get-started/introduction-to-iis/introduction-to-iis-architecturehttps://www.iis.net/learn/get-started/introduction-to-iis/introduction-to-iis-architecture>. Citado na página 38.

VALDRICH, P.; MISAGHI, M. Avaliação de desempenho de um sistema web por meio de testes automatizados. *Produção em Foco*, v. 5, n. 1, 2015. Citado na página 16.

VALENTINI, J.; GASPARY, L. P. Monitoração de disponibilidade e desempenho de servidores crıticos usando uma abordagem descentralizada. 2003. Citado na página 15.

VIEIRA, A. L. R. d. S. Administração e gerência de redes de computadores. Universidade Federal do Rio de Janeiro, 2010. Citado na página 24.

VIRTUALBOX. *Oracle VM VirtualBox*. s.d. Disponível em: <https://www.virtualbox.org/>. Citado na página 38.

WAZLAWICK, R. *Metodologia de pesquisa para ciência da computação*. [S.l.]: Elsevier Brasil, 2017. v. 2. Citado na página 34.

Xianghua Xu et al. Performance evaluation model of Web servers based on response time. In: *IEEE Conference Anthology*. IEEE, 2013. p. 1–5. ISBN 978-1-4799-1660-3. Disponível em: <http://ieeexplore.ieee.org/document/6784980/>. Citado nas páginas 16 e 33.

ZABBIX. *1.1 Introdução ao Zabbix [Zabbix Documentation 1.8]*. c2001–2020. Disponível em: <https://www.zabbix.com/documentation/1.8/pt/manual/sobre/introducao_ao_zabbix>. Citado na página 37.

ZHANG, L.; ZHU, Q. The Overtime Waiting Model for Web Server Performance Evaluation. In: *2014 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*. IEEE, 2014. p. 229–232. ISBN 978-1-4799-6236-5. Disponível em: <http://ieeexplore.ieee.org/document/6984311/>. Citado na página 16.

Apêndices

APÊNDICE A – Teste 1 - 100 Usuários

A.1 Dados Apache JMeter

Figura 13 – Teste 1 - Dados JMeter

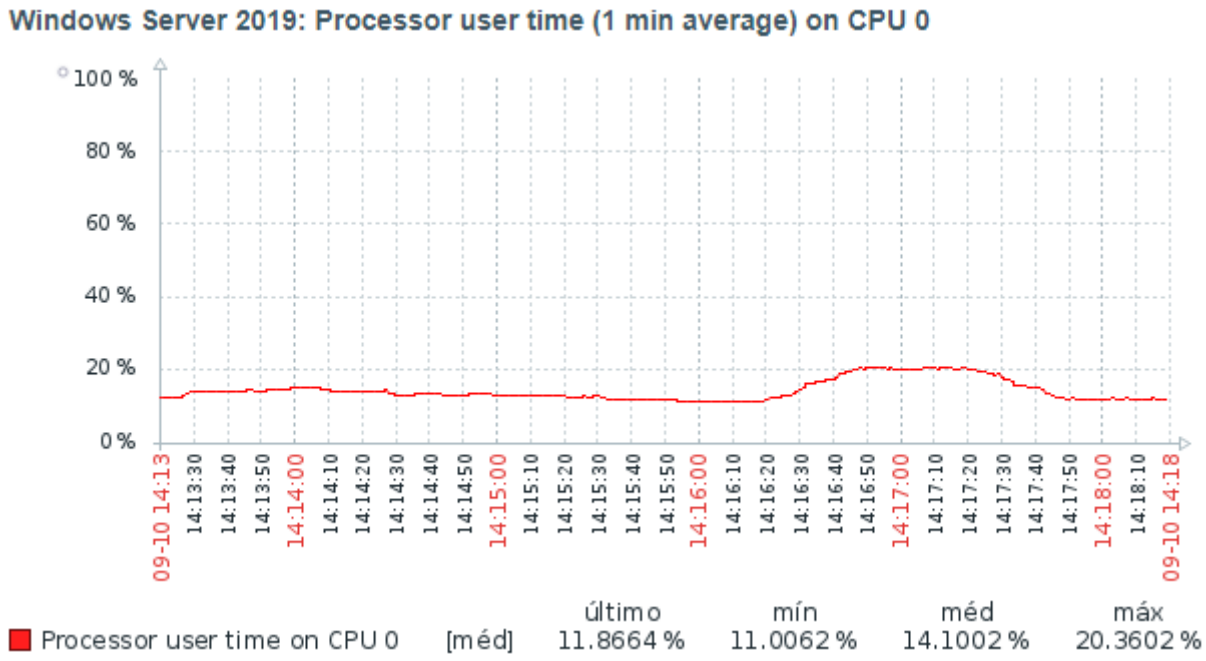
Rótulo	# Amostras	Média	Mín.	Máx.	Desvio Padrão	% de Erro	Vazão	KB/s	Sent KB/sec	Média de Bytes
login	100	54	29	142	21,87	0,00%	9,9/sec	416,25	19,01	43199,0
InserirRegistro	100	342	106	809	162,37	0,00%	5,6/sec	152,62	9,65	27710,0
ConsultarRegistr...	100	85	63	157	15,90	0,00%	9,3/sec	1926,78	4,54	211469,0
logout	100	44	24	90	14,47	0,00%	8,7/sec	26,93	12,58	3182,0
TOTAL	400	131	24	809	147,66	0,00%	12,2/sec	851,27	17,02	71390,0

Fonte: Elaborada pelo autor.

A.2 Dados Zabbix

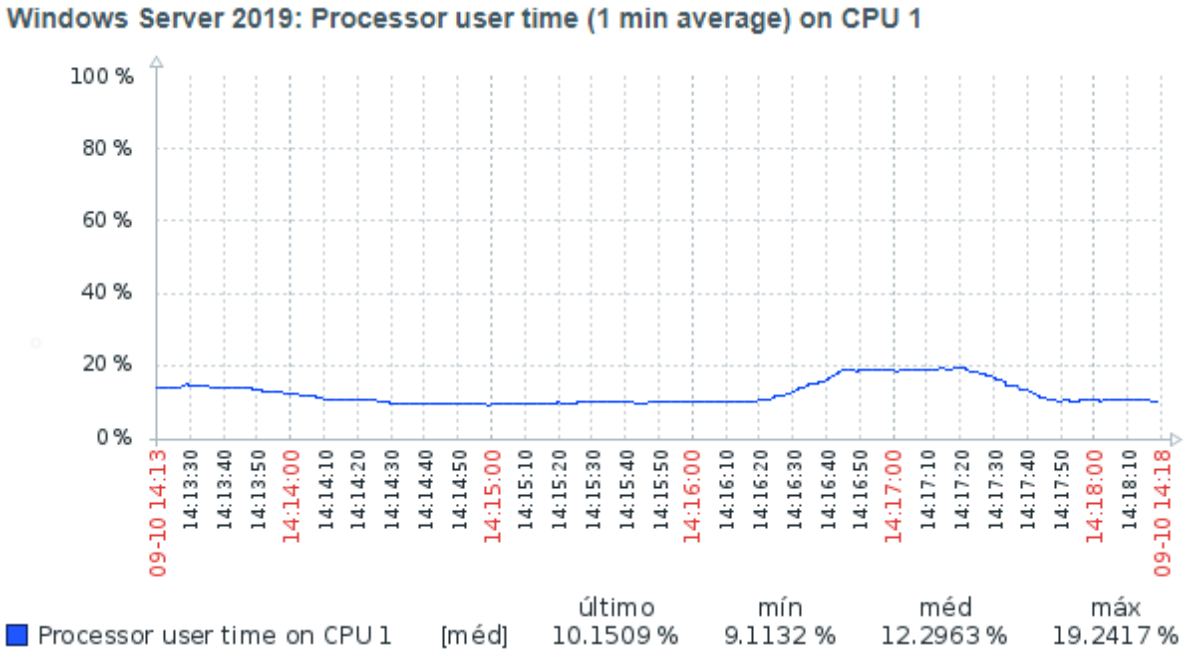
A.2.1 CPU

Figura 14 – Teste 1 - Percentual de utilização da CPU 0



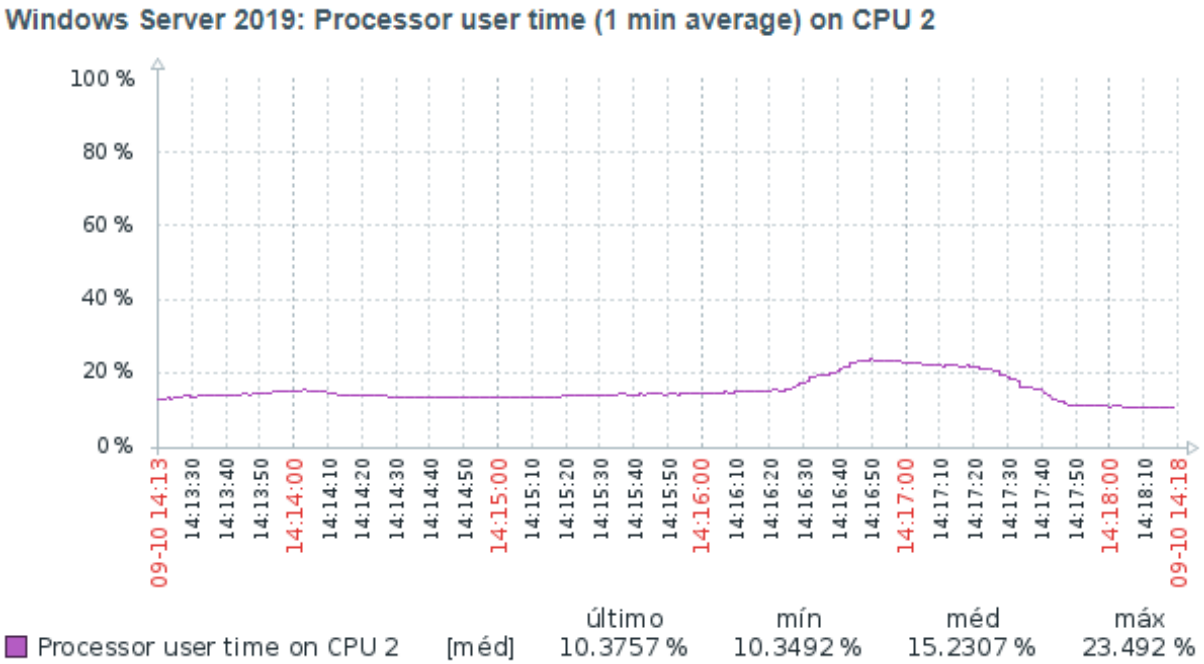
Fonte: Elaborada pelo autor.

Figura 15 – Teste 1 - Percentual de utilização da CPU 1



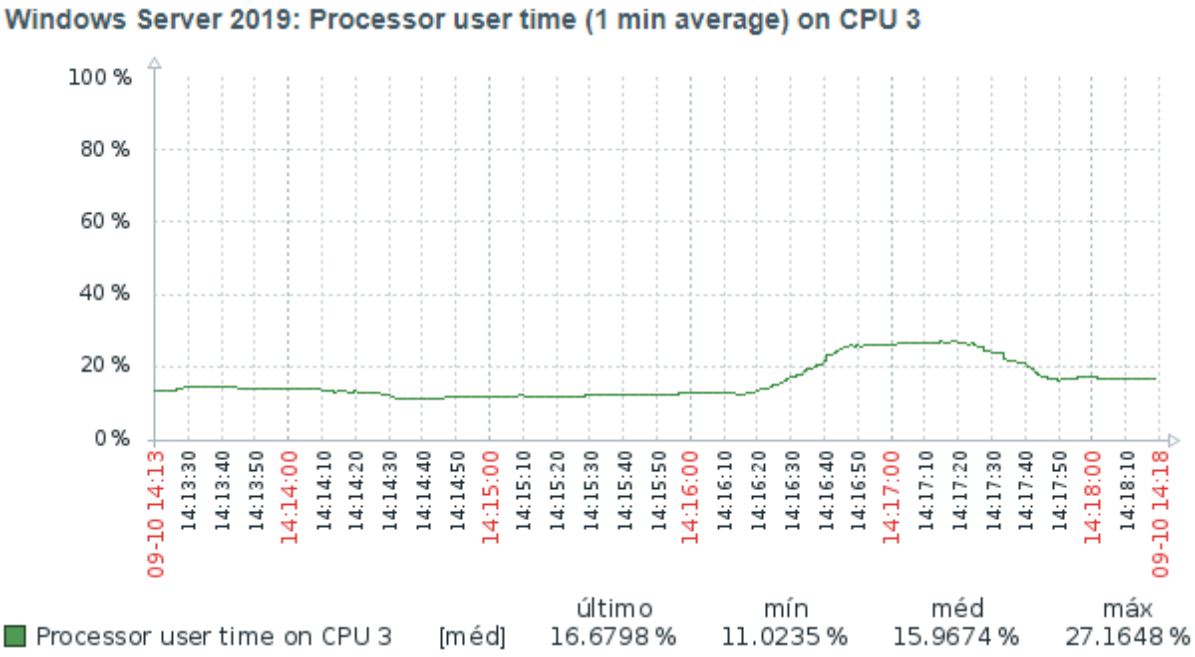
Fonte: Elaborada pelo autor.

Figura 16 – Teste 1 - Percentual de utilização da CPU 2



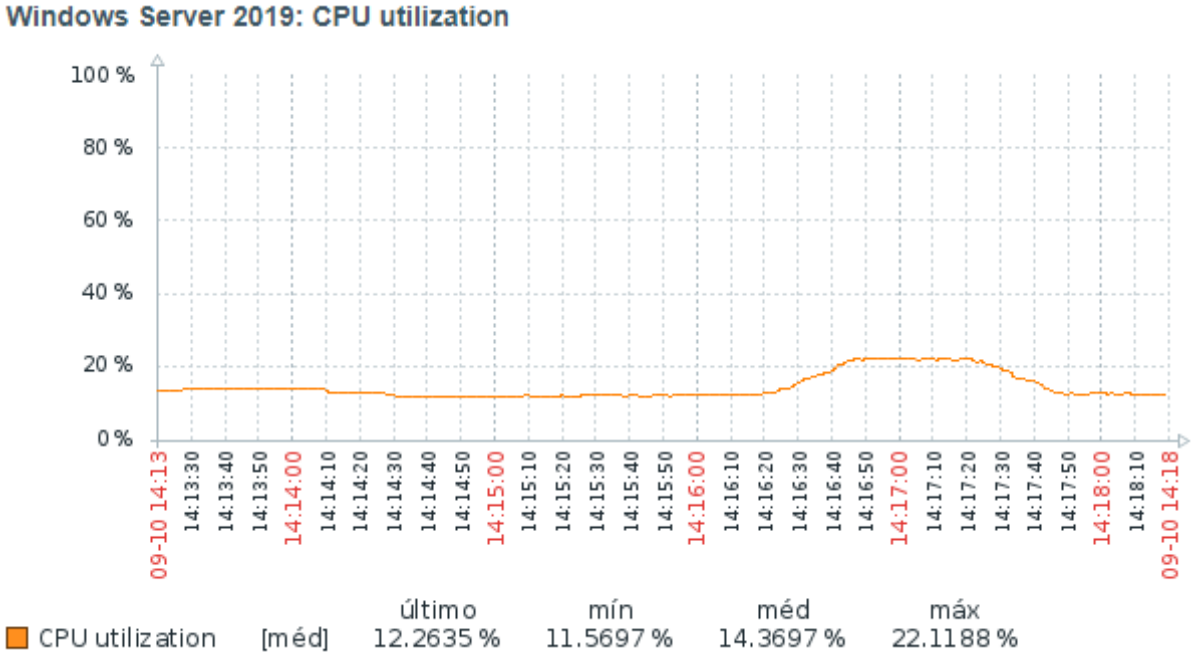
Fonte: Elaborada pelo autor.

Figura 17 – Teste 1 - Percentual de utilização da CPU 3



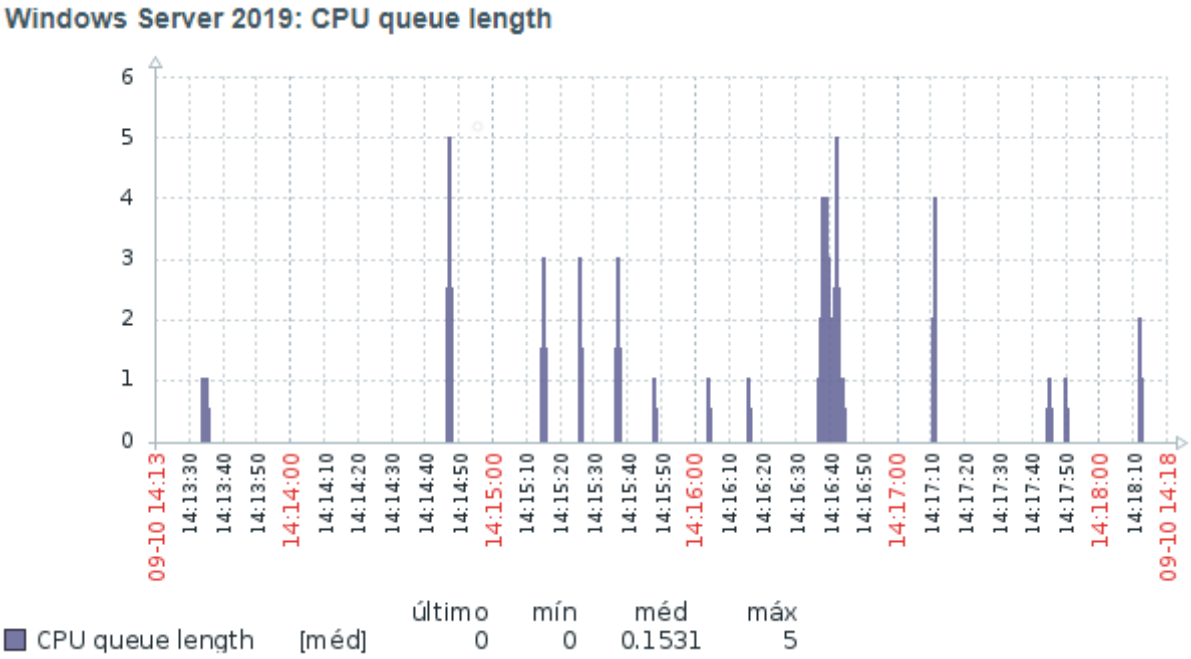
Fonte: Elaborada pelo autor.

Figura 18 – Teste 1 - Percentual de utilização da CPU Total



Fonte: Elaborada pelo autor.

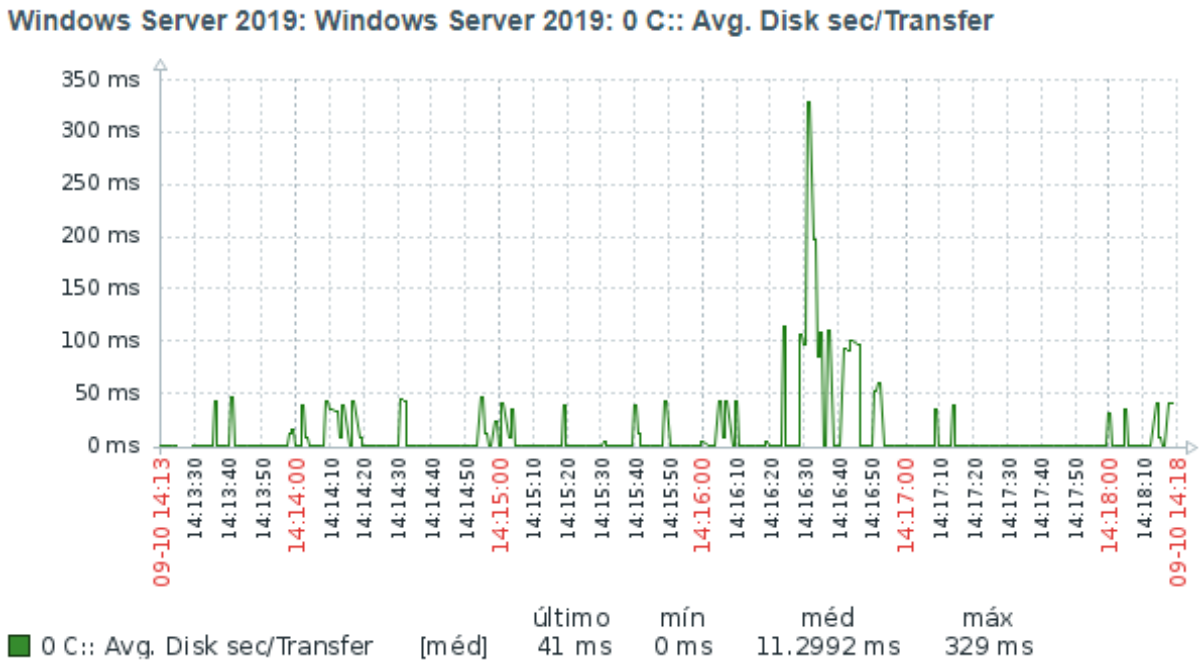
Figura 19 – Teste 1 - Fila de processo na CPU



Fonte: Elaborada pelo autor.

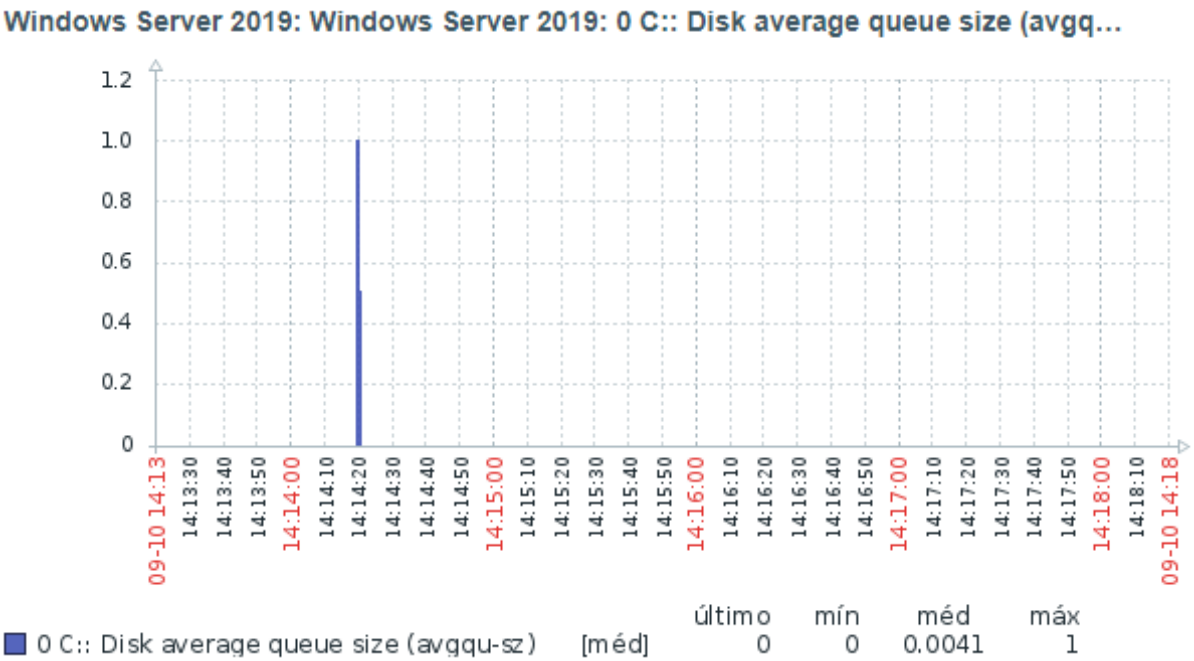
A.2.2 **Disco Rígido**

Figura 20 – Teste 1 - Tempo médio de leitura e escrita em disco



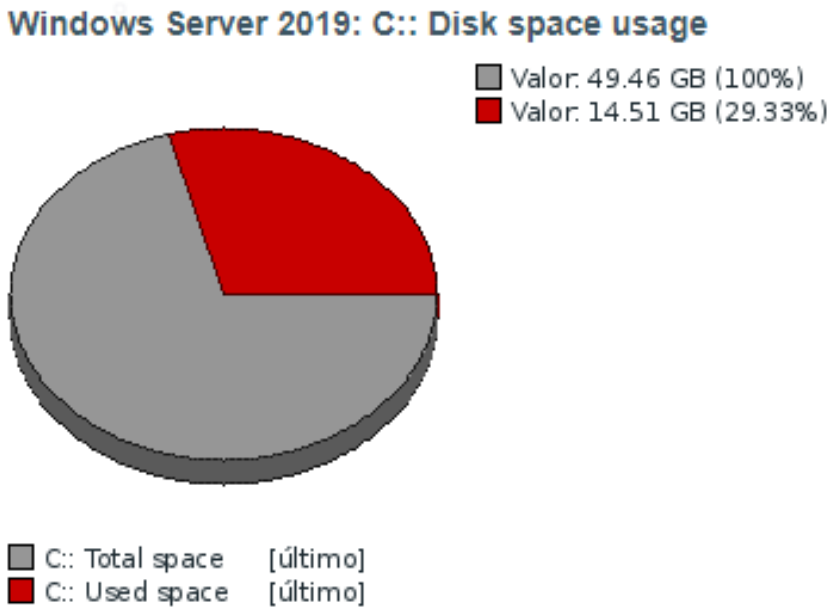
Fonte: Elaborada pelo autor.

Figura 21 – Teste 1 - Média da fila em disco



Fonte: Elaborada pelo autor.

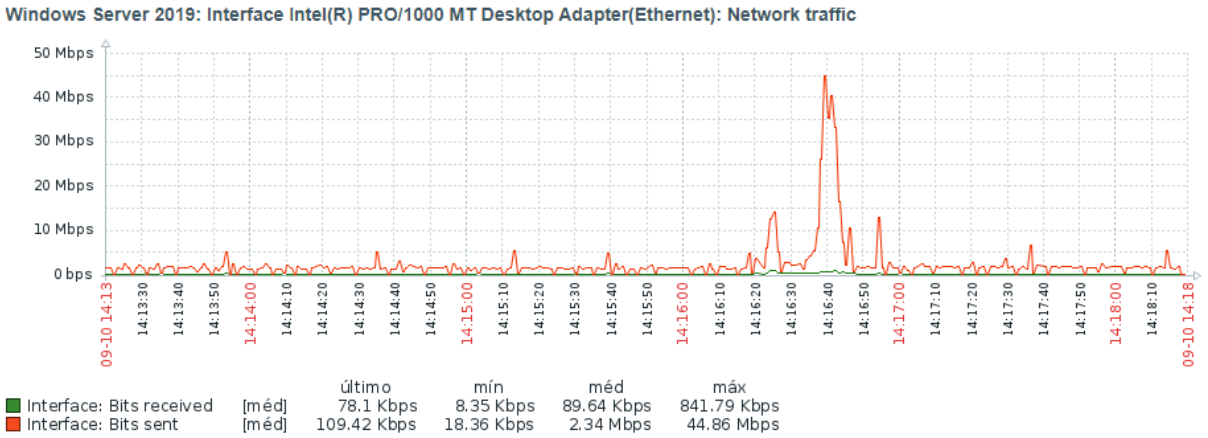
Figura 22 – Teste 1 - Espaço livre em disco



Fonte: Elaborada pelo autor.

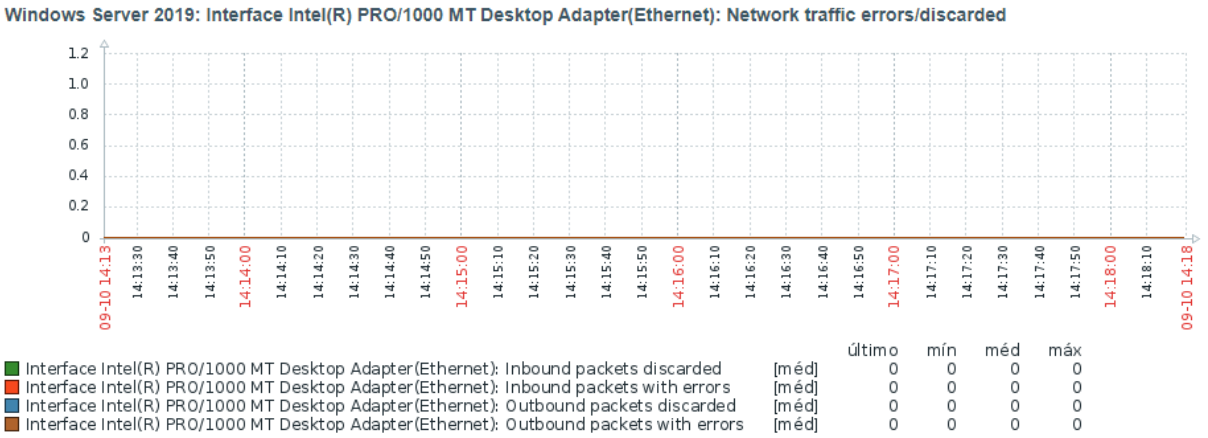
A.2.3 **Placa de rede**

Figura 23 – Teste 1 - Tráfego de rede



Fonte: Elaborada pelo autor.

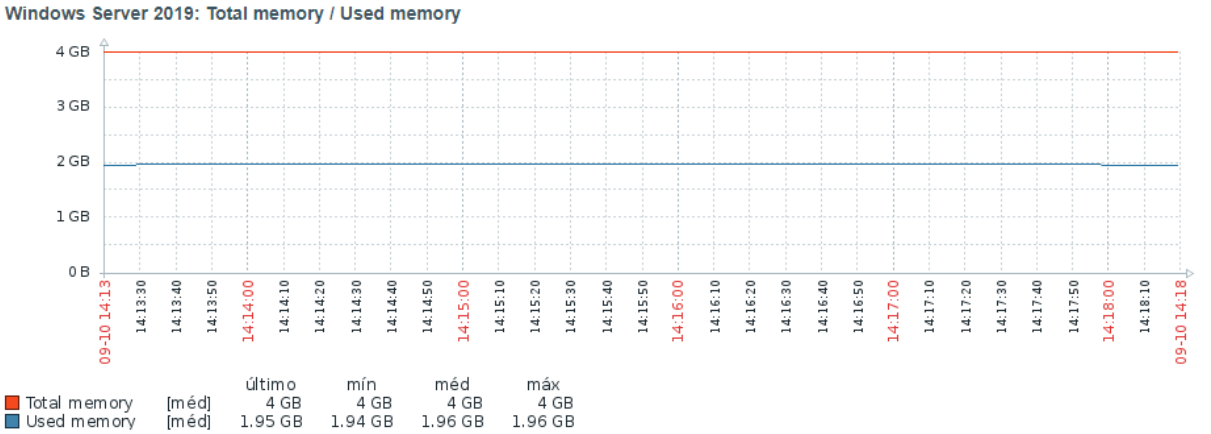
Figura 24 – Teste 1 - Descarte/Erros na Entrada/Saída de pacotes



Fonte: Elaborada pelo autor.

A.2.4 **Memória RAM**

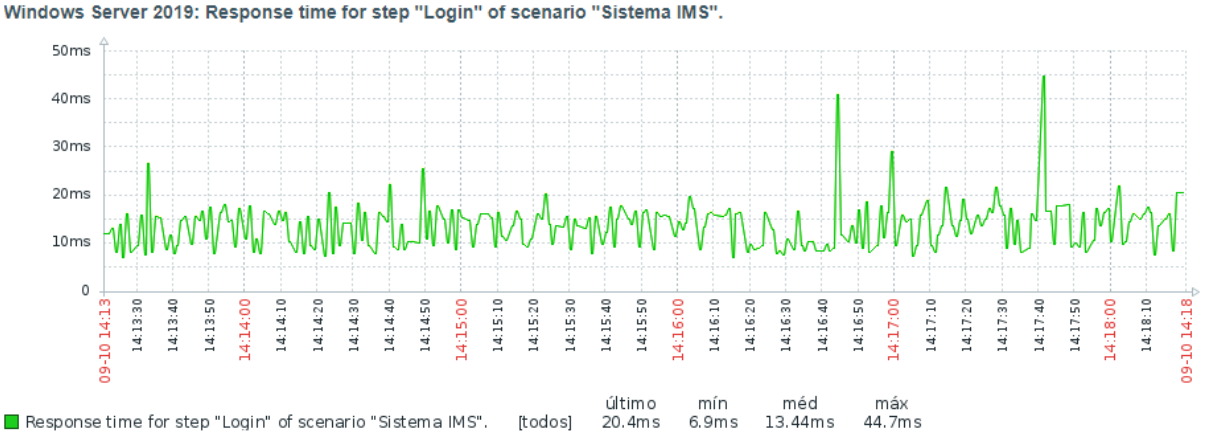
Figura 25 – Teste 1 - Total de memória e memória utilizada



Fonte: Elaborada pelo autor.

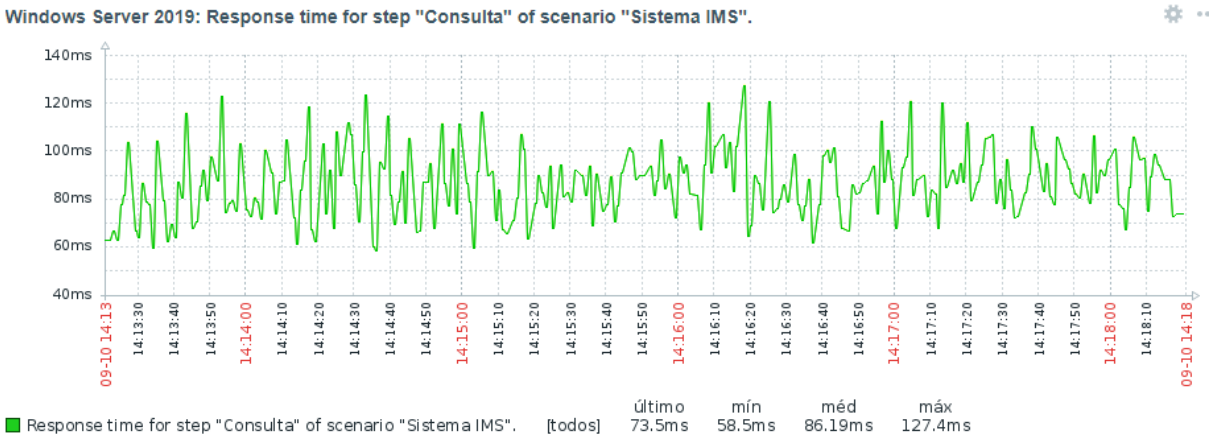
A.2.5 **Tempos de resposta**

Figura 26 – Teste 1 - Tempo de resposta login



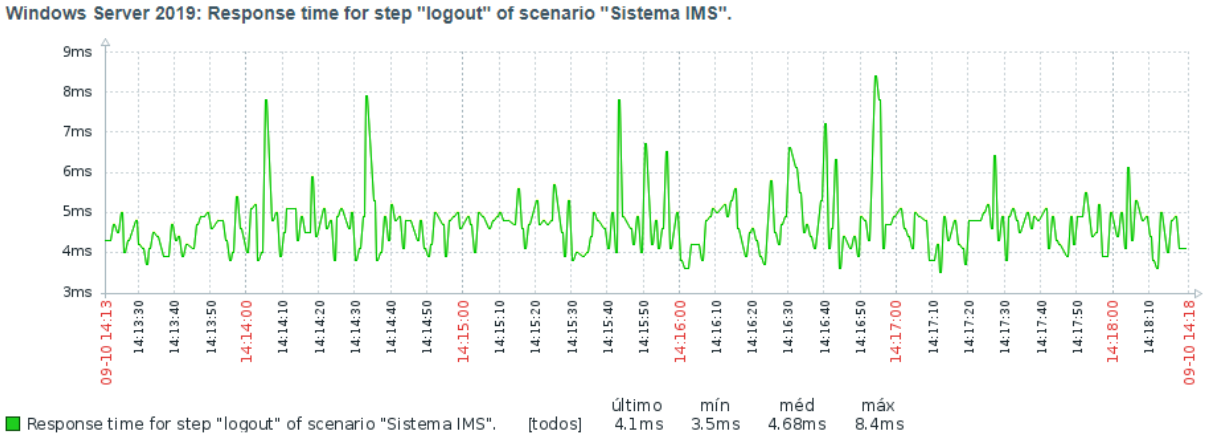
Fonte: Elaborada pelo autor.

Figura 27 – Teste 1 - Tempo de resposta consulta



Fonte: Elaborada pelo autor.

Figura 28 – Teste 1 - Tempo de resposta logout



Fonte: Elaborada pelo autor.

APÊNDICE B – Teste 2 - 200 Usuários

B.1 Dados Apache JMeter

Figura 29 – Teste 2 - Dados JMeter

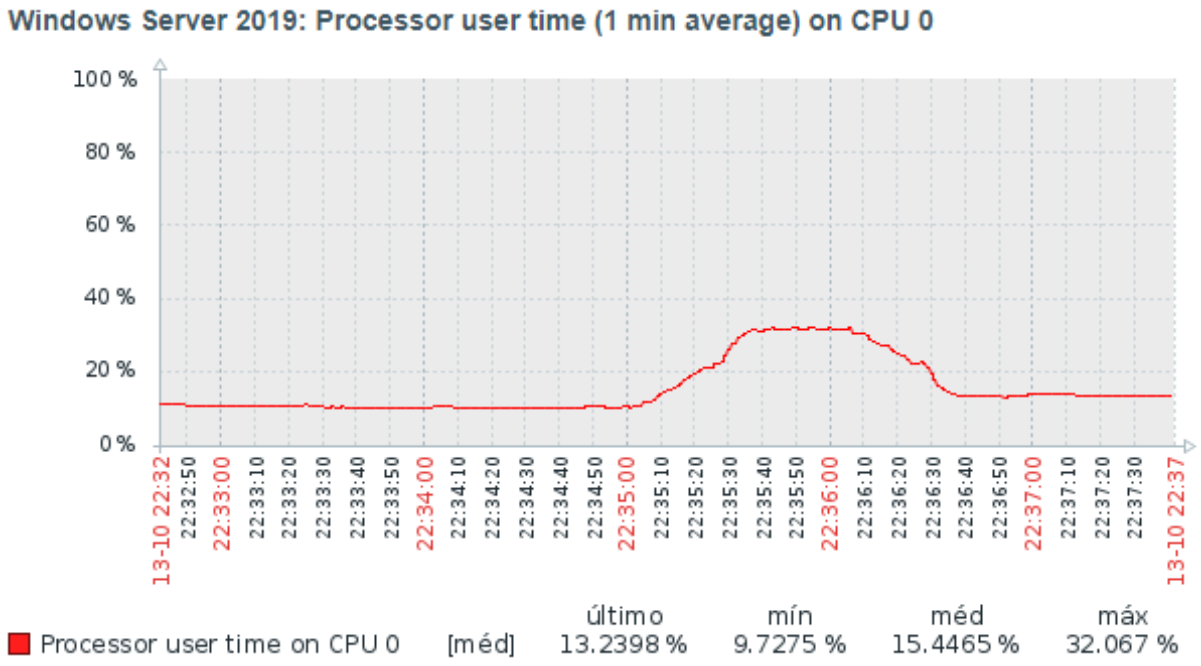
Rótulo	# Amostras	Média	Mín.	Máx.	Desvio Padrão	% de Erro	Vazão	KB/s	Sent KB/sec	Média de Bytes
login	200	132	29	380	102,65	0,00%	18,6/sec	786,40	35,92	43199,0
InserirRegistro	200	3821	2590	4881	532,08	0,00%	9,5/sec	257,63	16,30	27710,0
ConsultarRegistr...	200	669	84	1763	450,69	0,00%	16,6/sec	4559,06	8,07	281369,0
logout	200	837	23	3945	1124,87	0,00%	15,3/sec	47,62	22,25	3182,0
TOTAL	800	1365	23	4881	1587,19	0,00%	21,6/sec	1873,18	30,09	88865,0

Fonte: Elaborada pelo autor.

B.2 **Dados Zabbix**

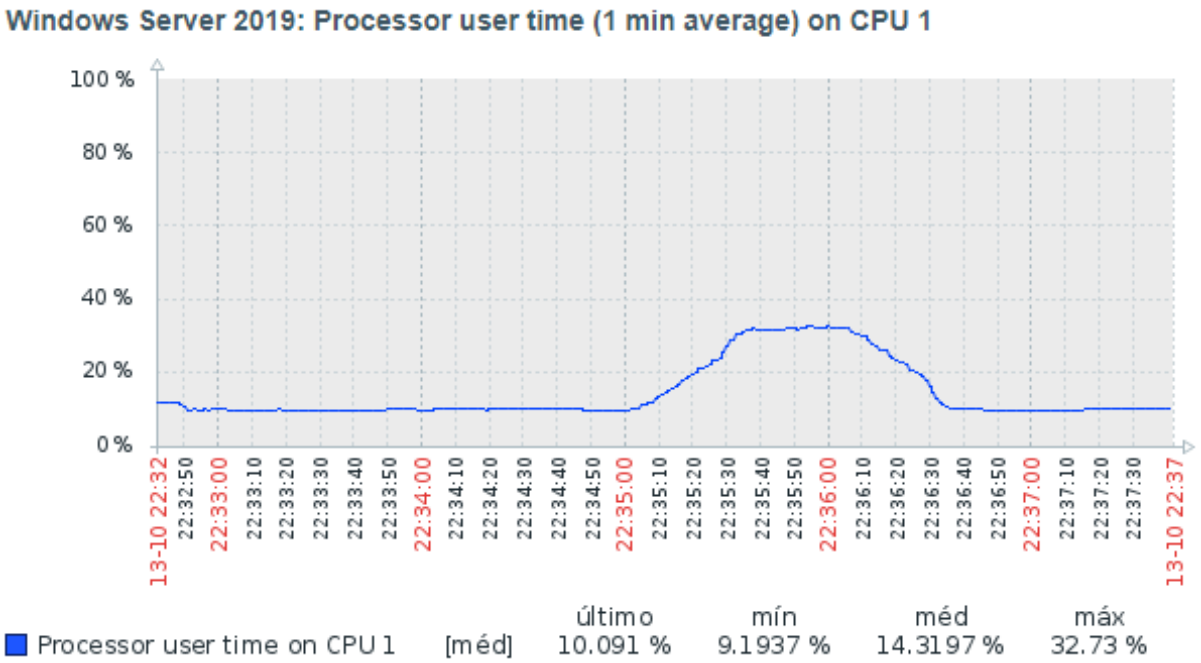
B.2.1 **CPU**

Figura 30 – Teste 2 - Percentual de utilização da CPU 0



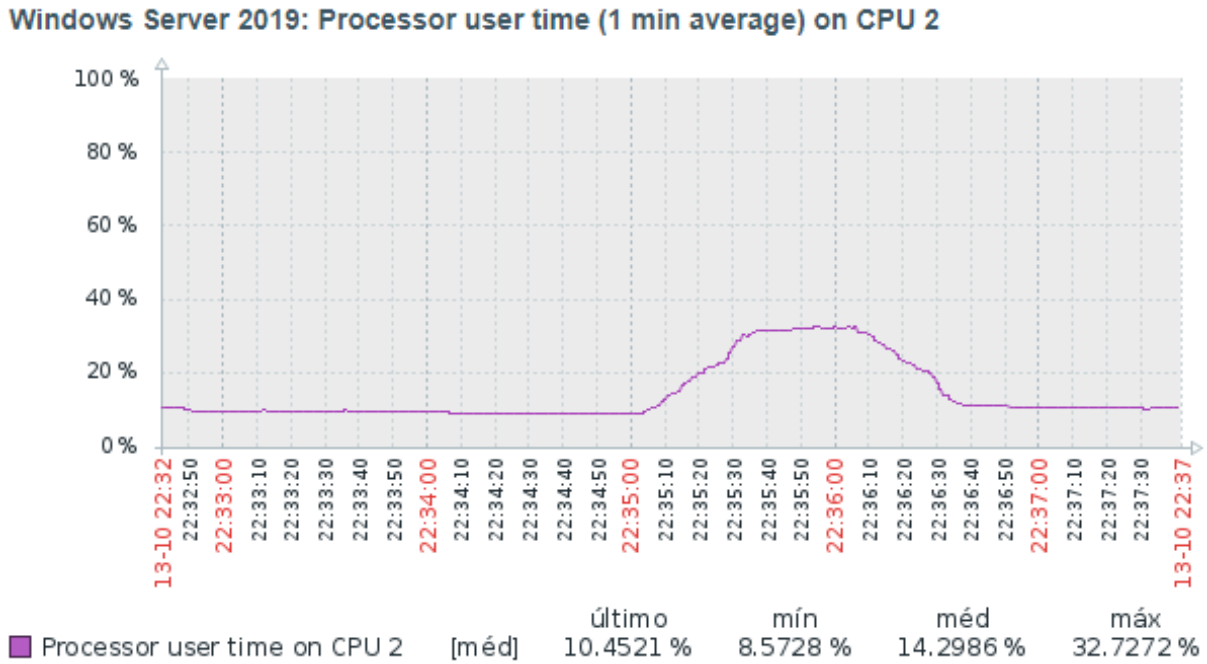
Fonte: Elaborada pelo autor.

Figura 31 – Teste 2 - Percentual de utilização da CPU 1



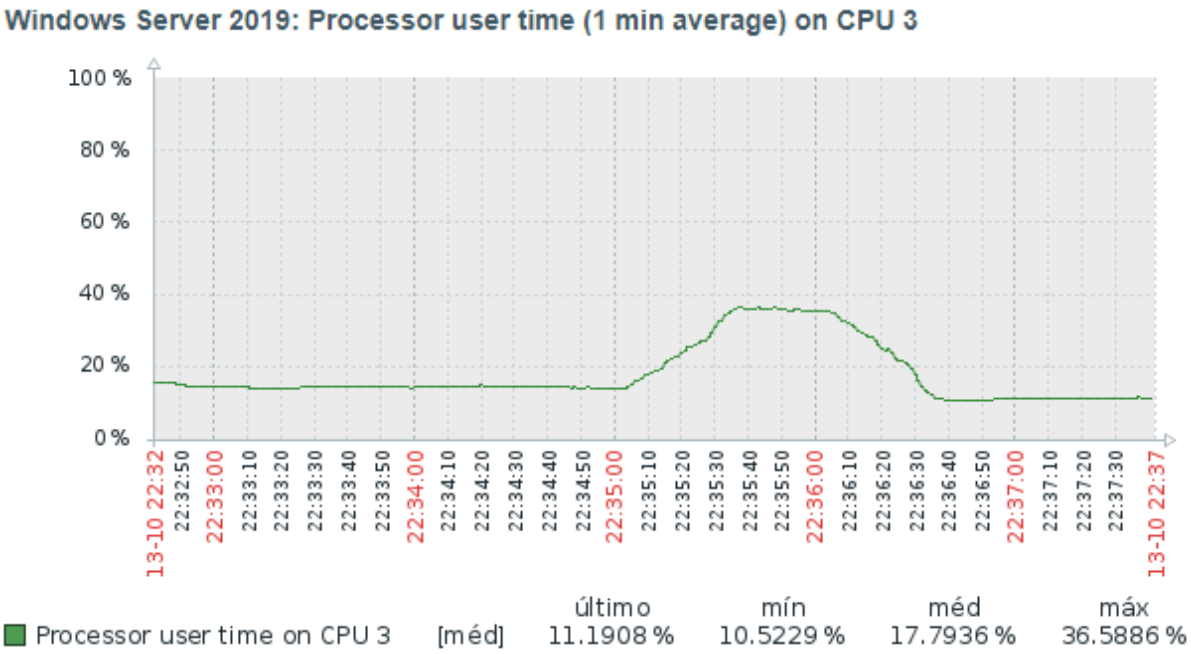
Fonte: Elaborada pelo autor.

Figura 32 – Teste 2 - Percentual de utilização da CPU 2



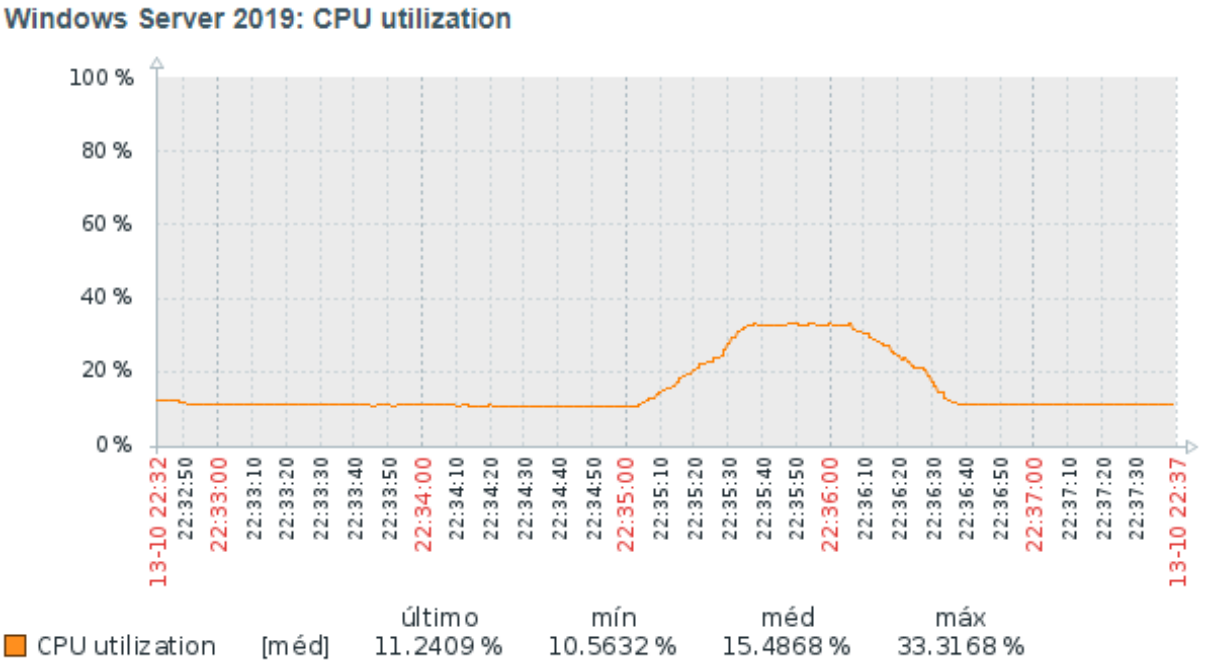
Fonte: Elaborada pelo autor.

Figura 33 – Teste 2 - Percentual de utilização da CPU 3



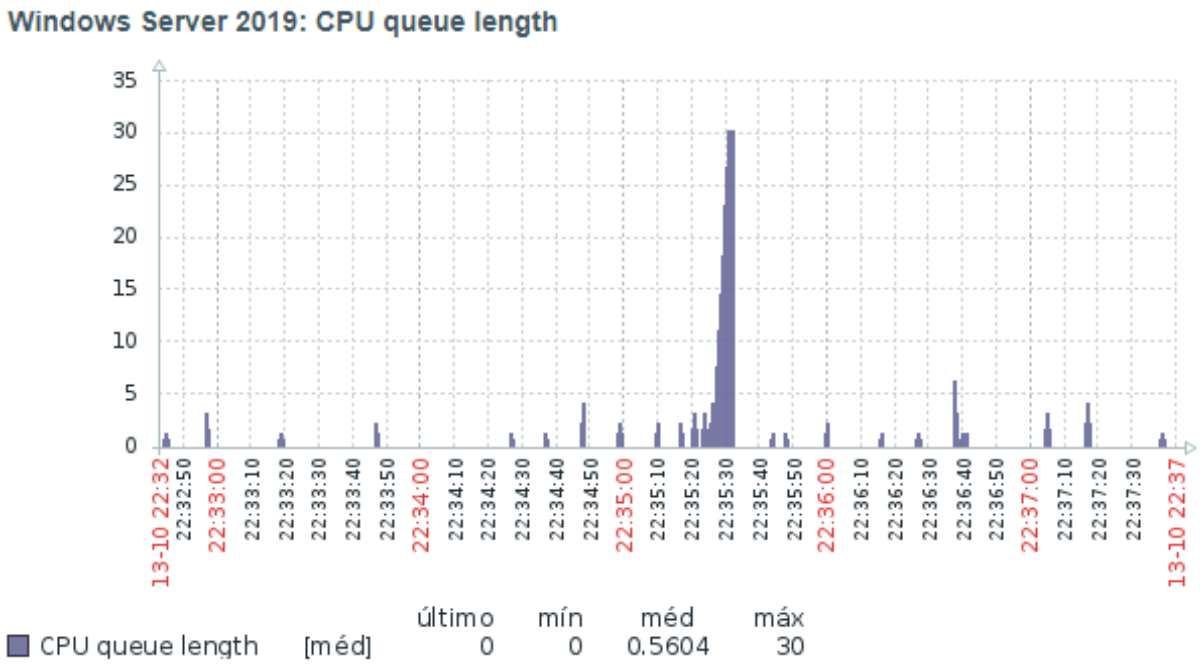
Fonte: Elaborada pelo autor.

Figura 34 – Teste 2 - Percentual de utilização da CPU Total



Fonte: Elaborada pelo autor.

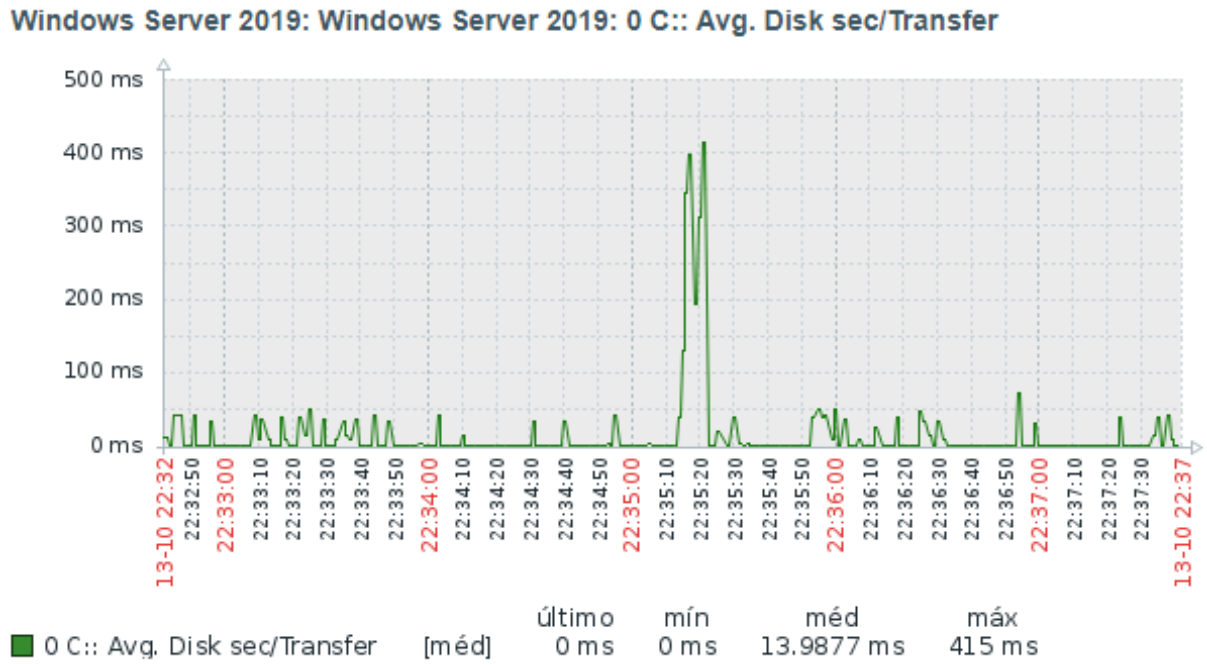
Figura 35 – Teste 2 - Fila de processo na CPU



Fonte: Elaborada pelo autor.

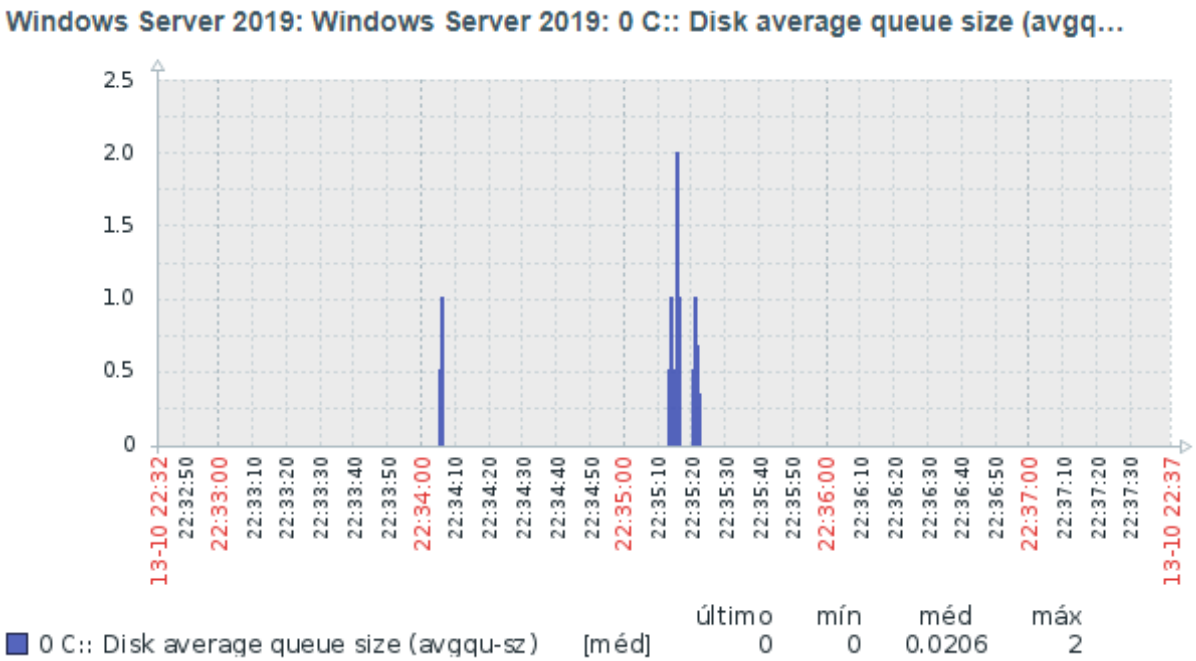
B.2.2 **Disco Rígido**

Figura 36 – Teste 2 - Tempo médio de leitura e escrita em disco



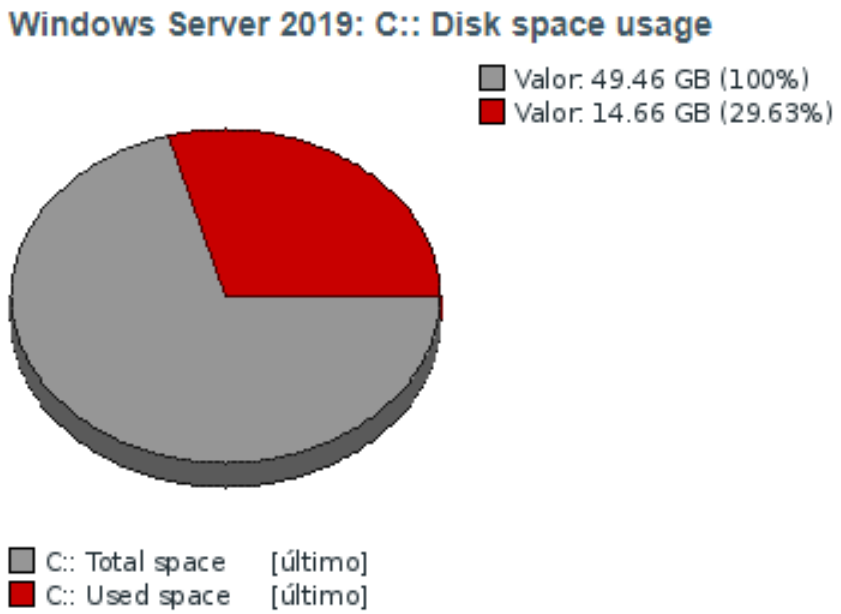
Fonte: Elaborada pelo autor.

Figura 37 – Teste 2 - Média da fila em disco



Fonte: Elaborada pelo autor.

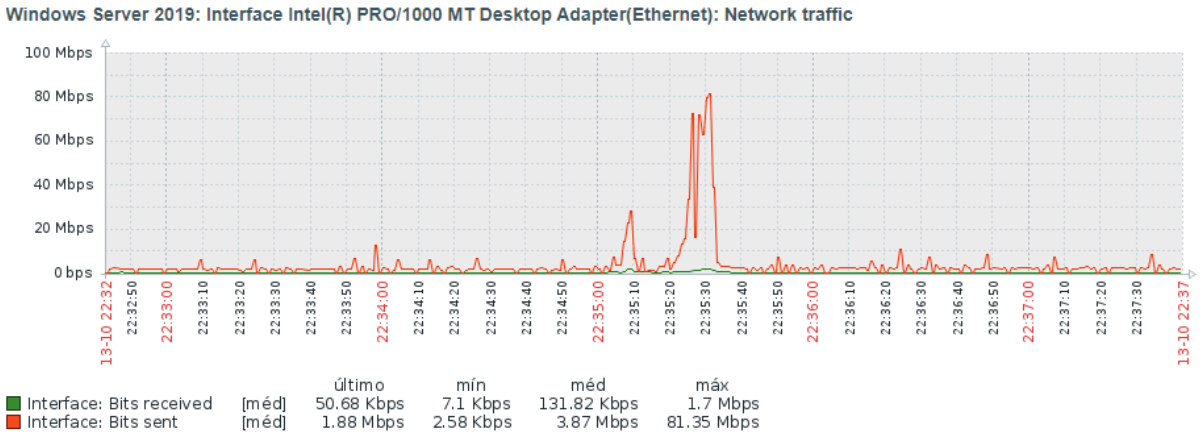
Figura 38 – Teste 2 - Espaço livre em disco



Fonte: Elaborada pelo autor.

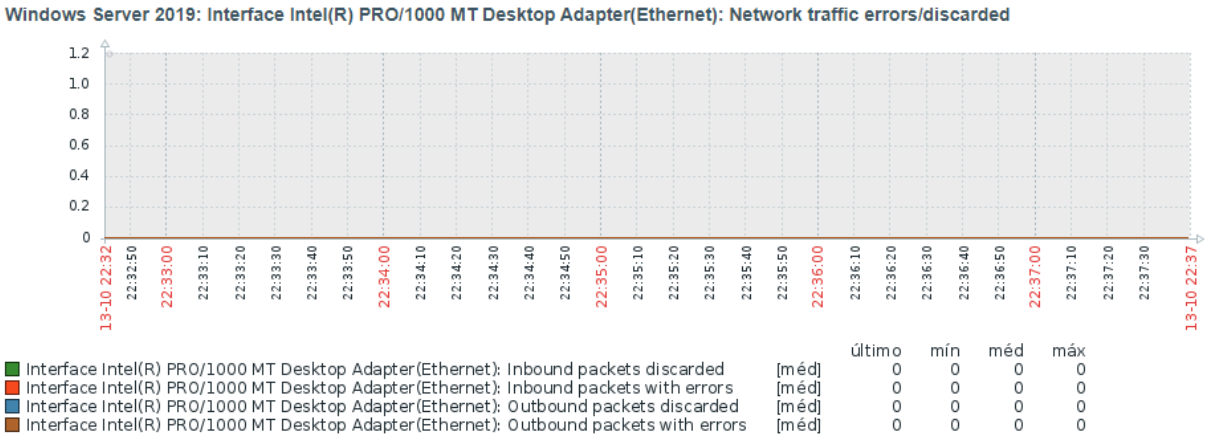
B.2.2.1 **Placa de rede**

Figura 39 – Teste 2 - Tráfego de rede



Fonte: Elaborada pelo autor.

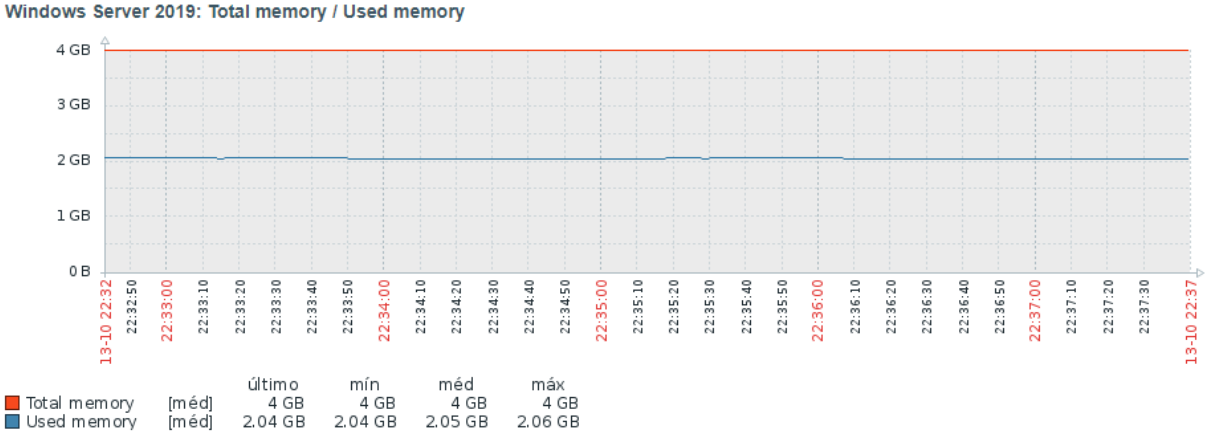
Figura 40 – Teste 2 - Descarte/Erros na Entrada/Saída de pacotes



Fonte: Elaborada pelo autor.

B.2.3 **Memória RAM**

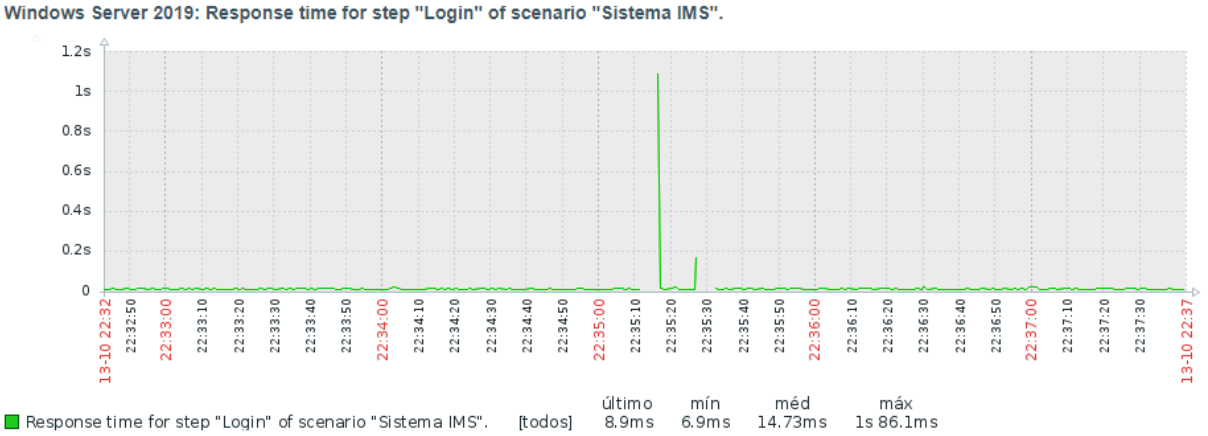
Figura 41 – Teste 2 - Total de memória e memória utilizada



Fonte: Elaborada pelo autor.

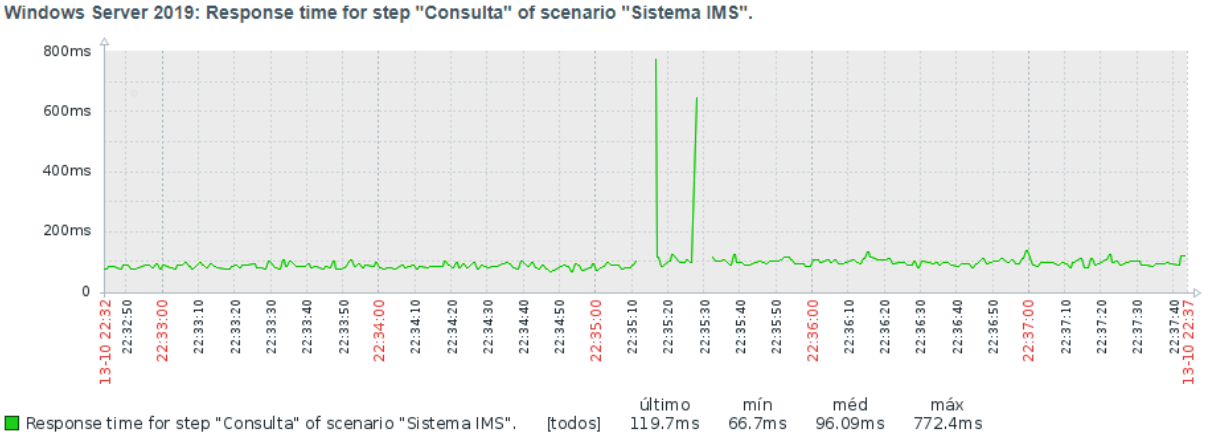
B.2.4 **Tempos de resposta**

Figura 42 – Teste 2 - Tempo de resposta login



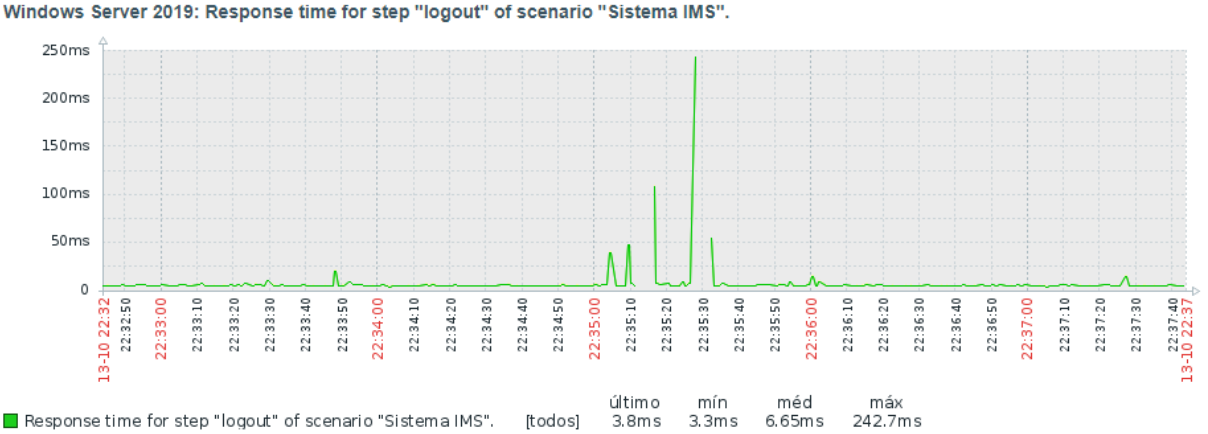
Fonte: Elaborada pelo autor.

Figura 43 – Teste 2 - Tempo de resposta consulta



Fonte: Elaborada pelo autor.

Figura 44 – Teste 2 - Tempo de resposta logout



Fonte: Elaborada pelo autor.

APÊNDICE C – Teste 3 - 300 Usuários

C.1 Dados Apache JMeter

Figura 45 – Teste 3 - Dados JMeter

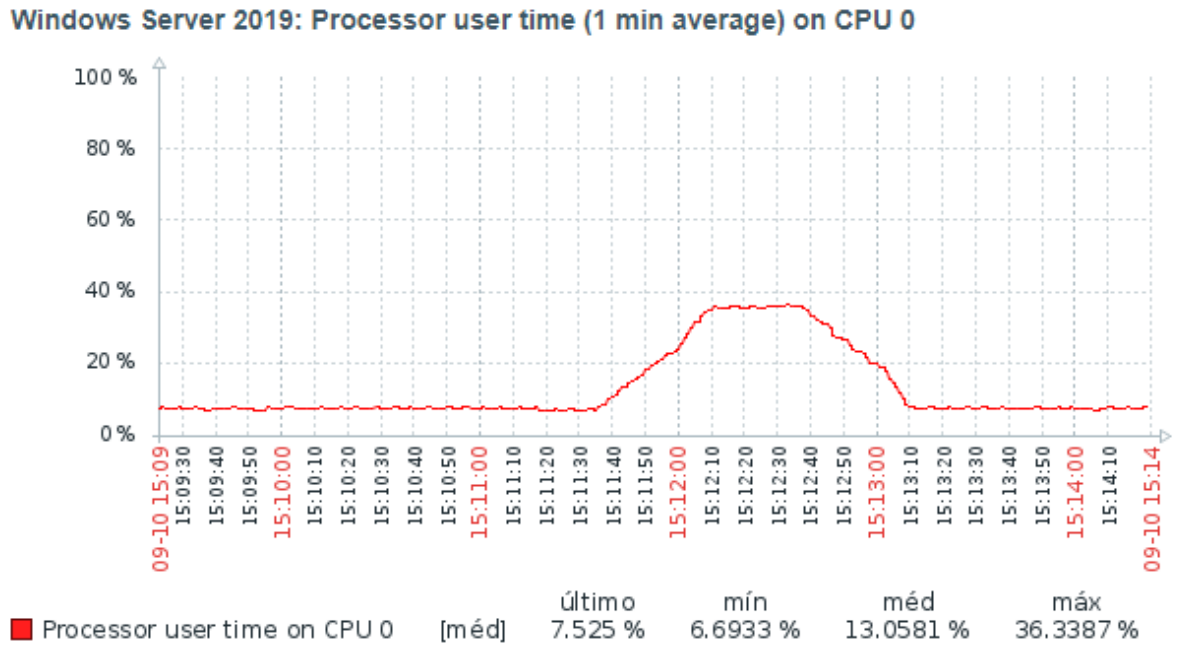
Rótulo	# Amostras	Média	Mín.	Máx.	Desvio Padrão	% de Erro	Vazão	KB/s	Sent KB/sec	Média de Bytes
login	300	1407	26	2433	731,53	0,00%	24,2/sec	1022,62	46,71	43199,0
InserirRegistro	300	3757	2616	5141	551,54	0,00%	13,4/sec	363,59	23,00	27710,0
ConsultarRegistr...	300	939	74	2536	531,71	0,00%	19,3/sec	6194,90	9,41	327962,8
logout	300	1055	17	4383	1247,92	0,00%	19,4/sec	60,40	28,22	3182,0
TOTAL	1200	1790	17	5141	1410,52	0,00%	30,3/sec	2973,95	42,24	100513,4

Fonte: Elaborada pelo autor.

C.2 **Dados Zabbix**

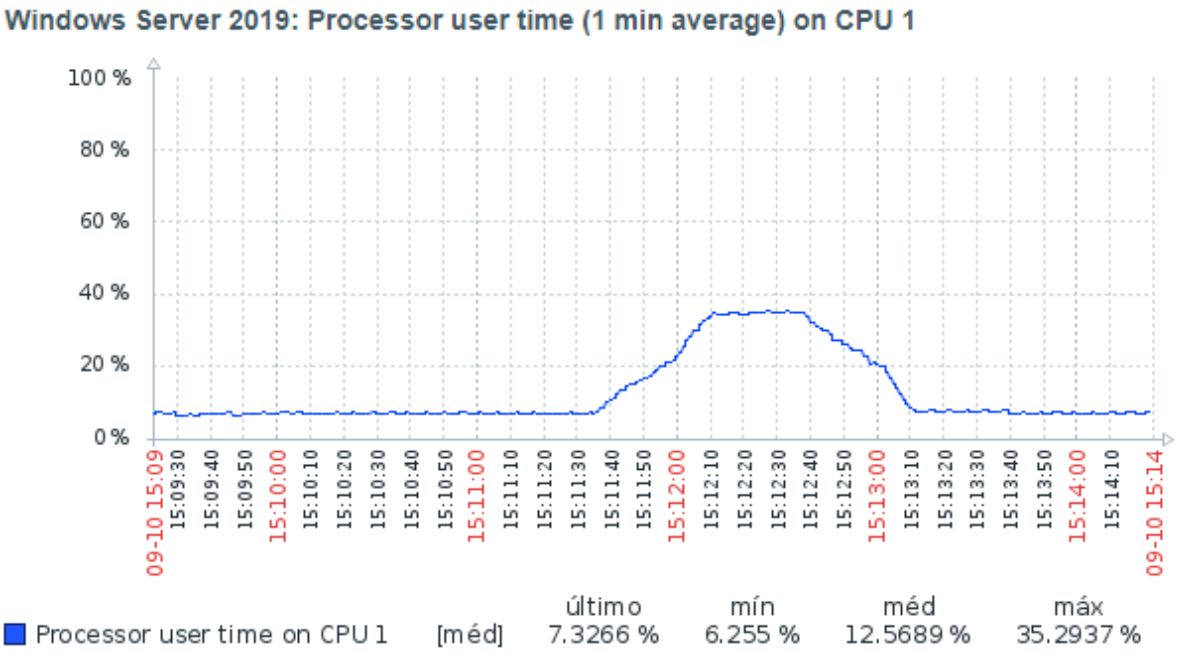
C.2.1 **CPU**

Figura 46 – Teste 3 - Percentual de utilização da CPU 0



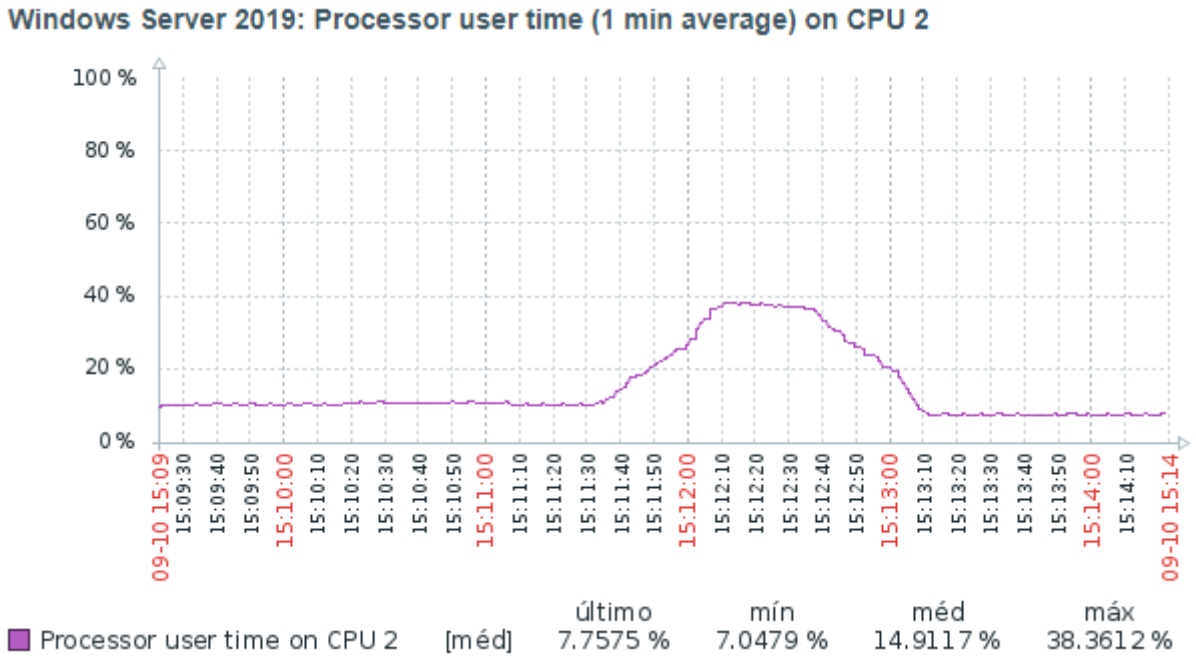
Fonte: Elaborada pelo autor.

Figura 47 – Teste 3 - Percentual de utilização da CPU 1



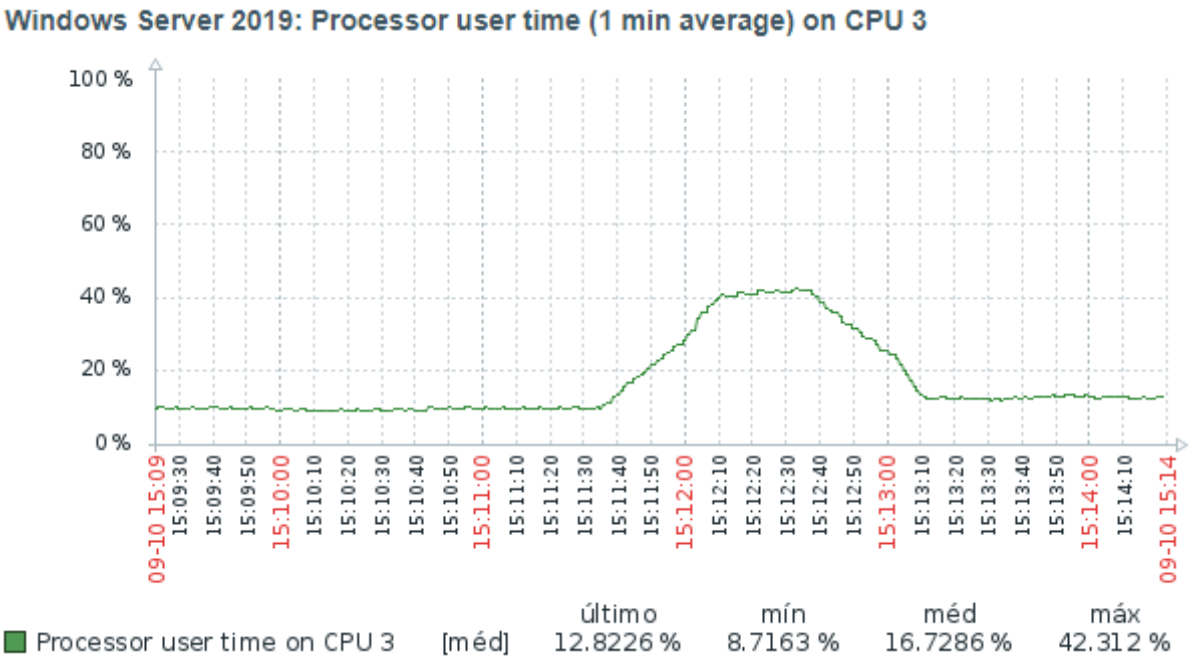
Fonte: Elaborada pelo autor.

Figura 48 – Teste 3 - Percentual de utilização da CPU 2



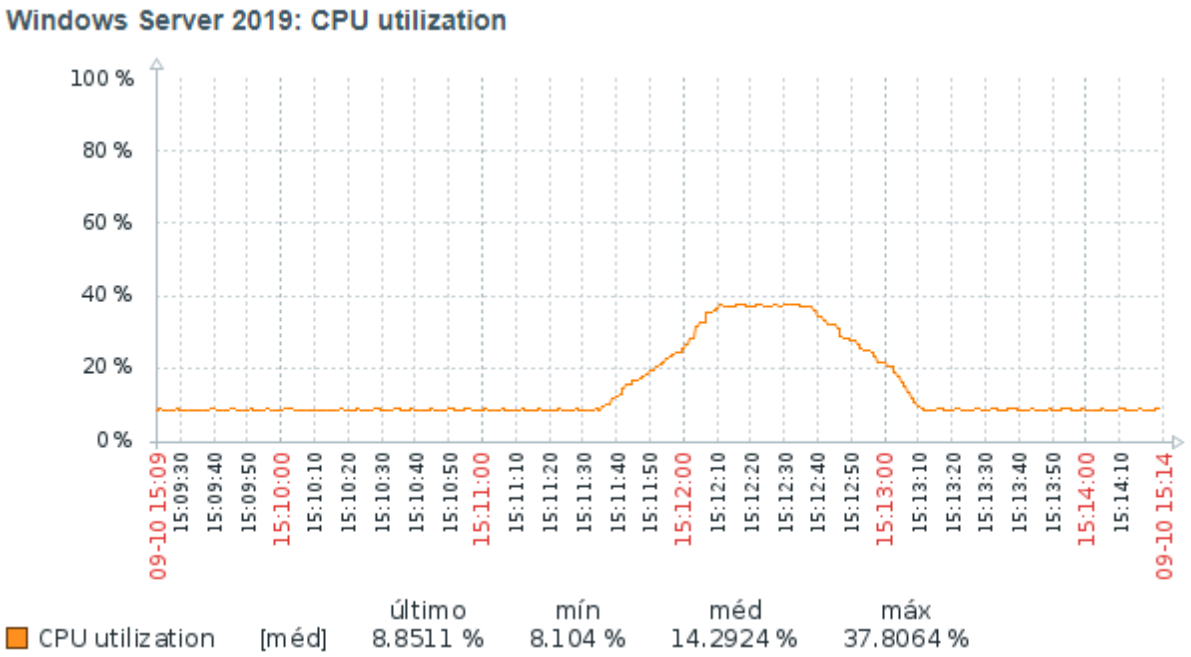
Fonte: Elaborada pelo autor.

Figura 49 – Teste 3 - Percentual de utilização da CPU 3



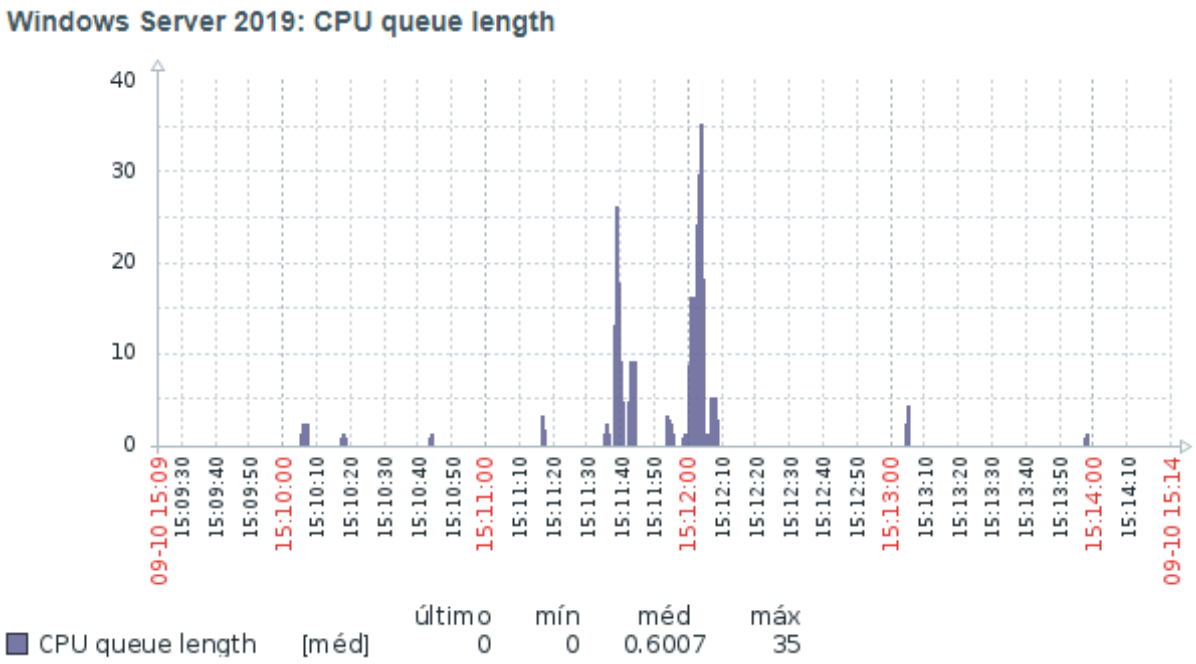
Fonte: Elaborada pelo autor.

Figura 50 – Teste 3 - Percentual de utilização da CPU Total



Fonte: Elaborada pelo autor.

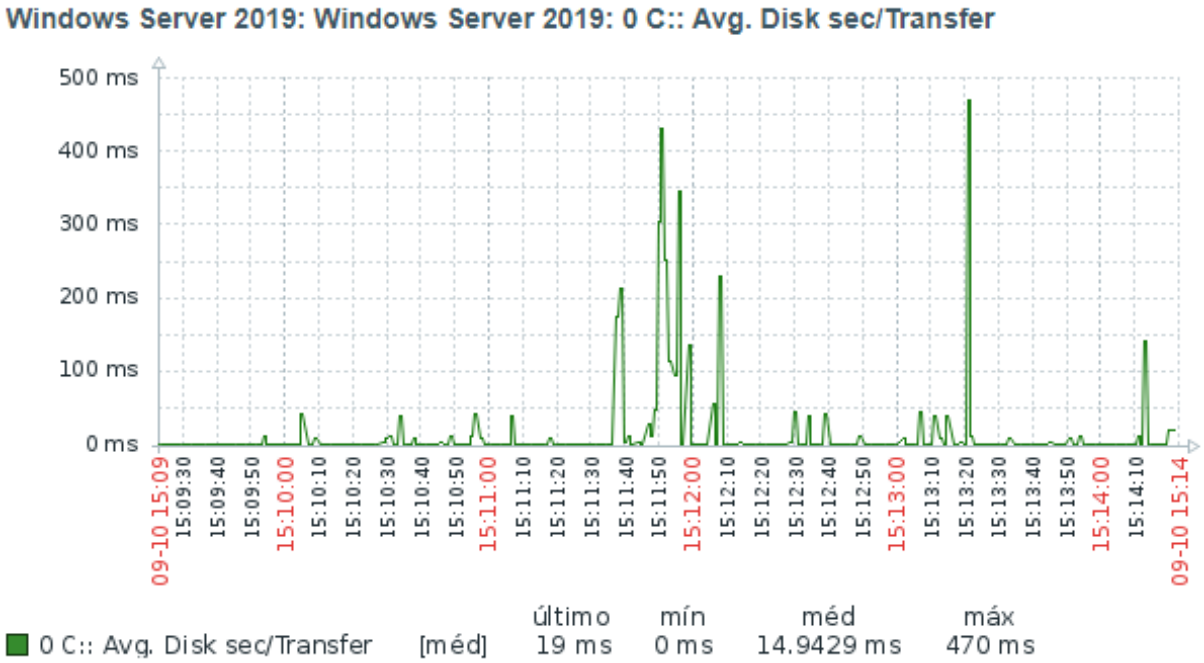
Figura 51 – Teste 3 - Fila de processo na CPU



Fonte: Elaborada pelo autor.

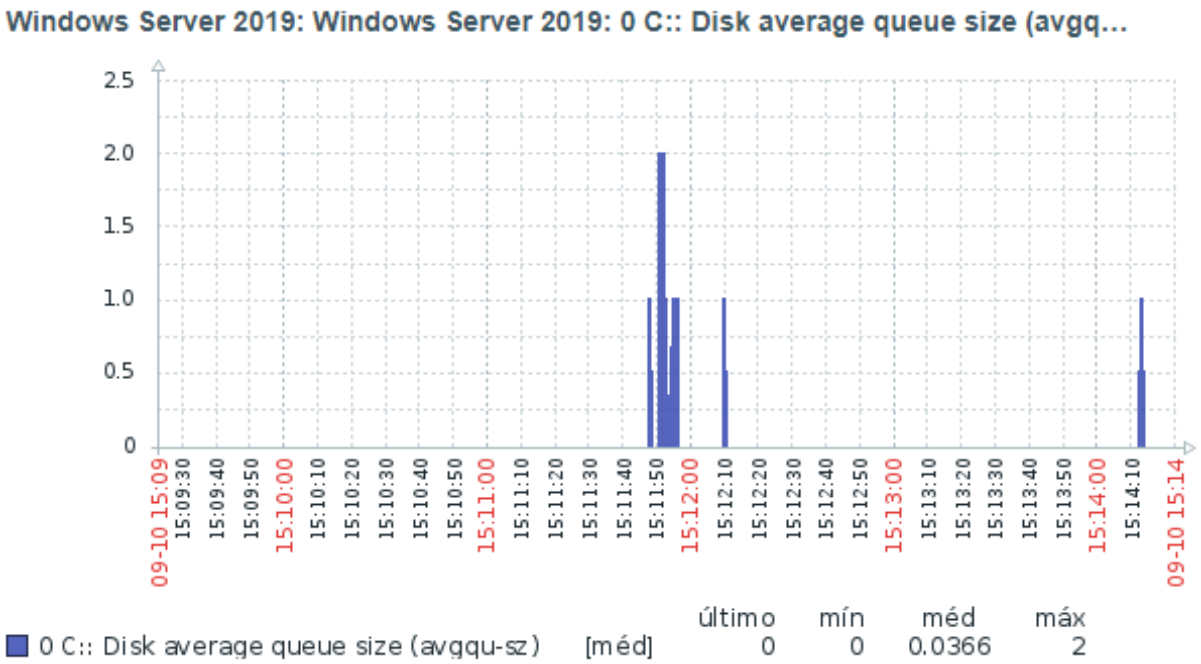
C.2.2 **Disco Rígido**

Figura 52 – Teste 3 - Tempo médio de leitura e escrita por transferência em disco



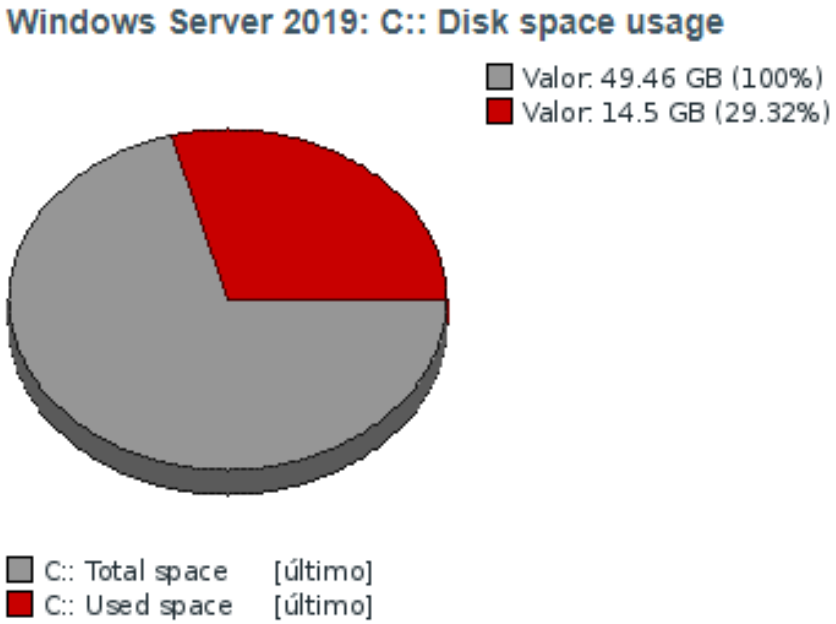
Fonte: Elaborada pelo autor.

Figura 53 – Teste 3 - Média da fila em disco



Fonte: Elaborada pelo autor.

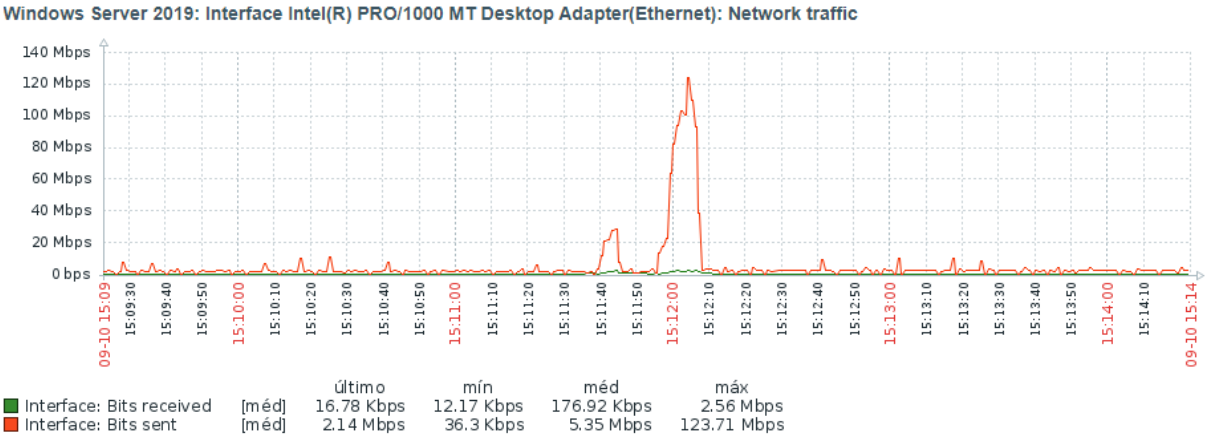
Figura 54 – Teste 3 - Espaço livre em disco



Fonte: Elaborada pelo autor.

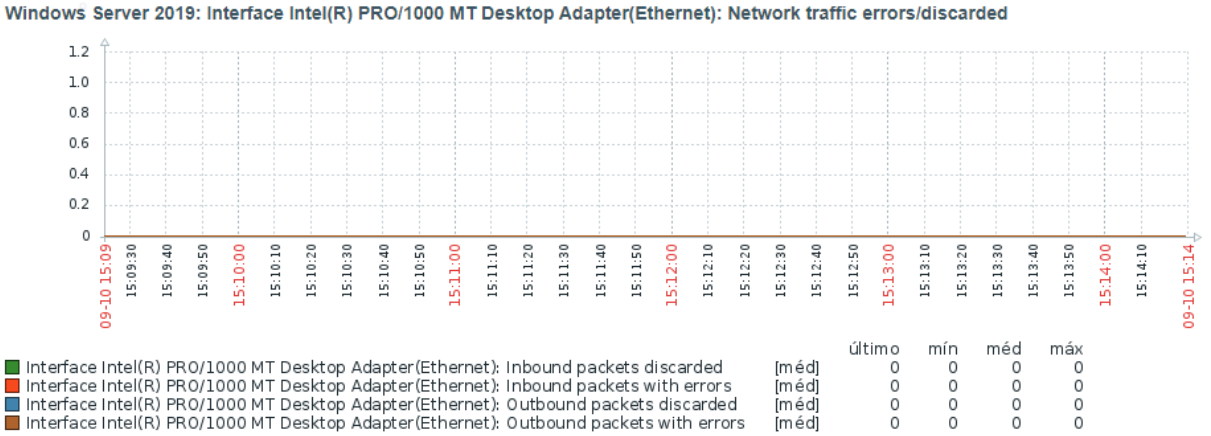
C.2.3 **Placa de rede**

Figura 55 – Teste 3 - Tráfego de rede



Fonte: Elaborada pelo autor.

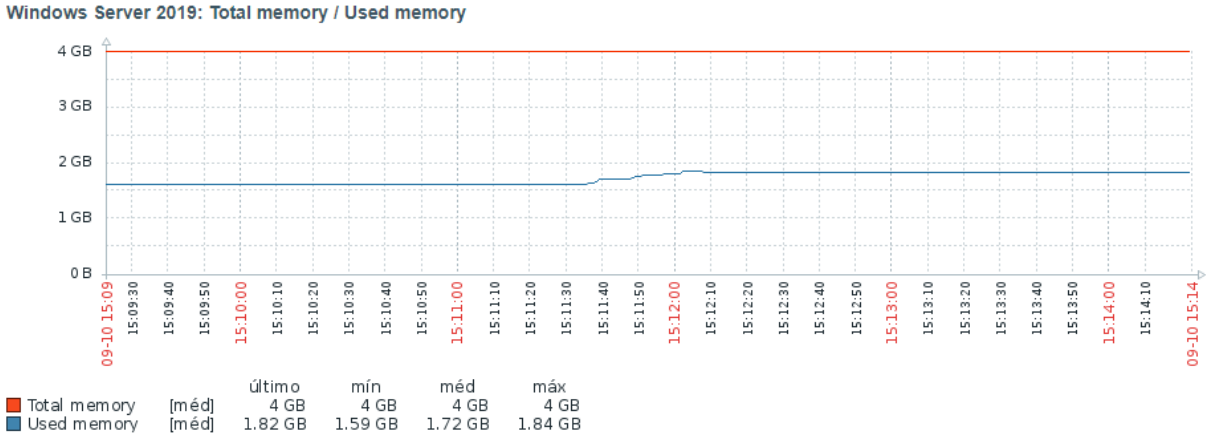
Figura 56 – Teste 3 - Descarte/Erros na Entrada/Saída de pacotes



Fonte: Elaborada pelo autor.

C.2.4 **Memória RAM**

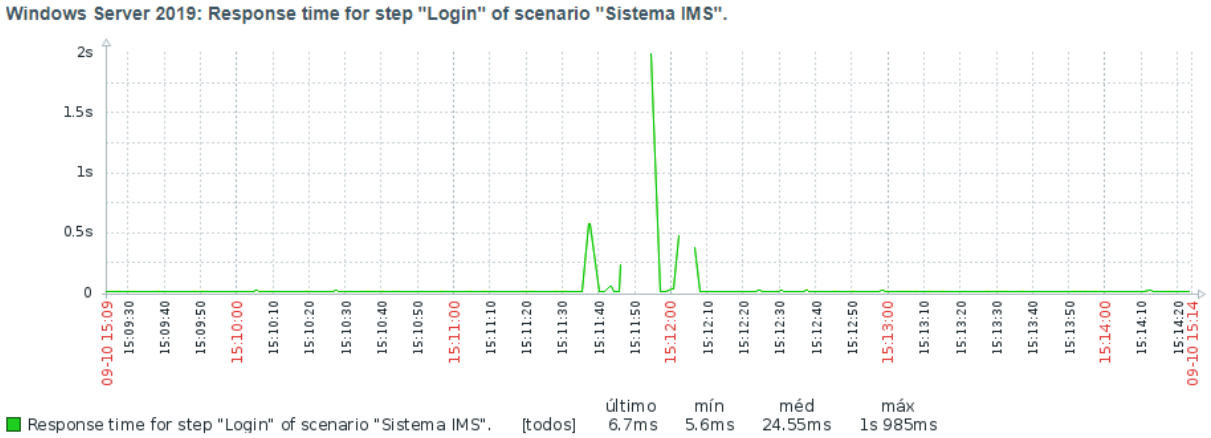
Figura 57 – Teste 3 - Total de memória e memória utilizada



Fonte: Elaborada pelo autor.

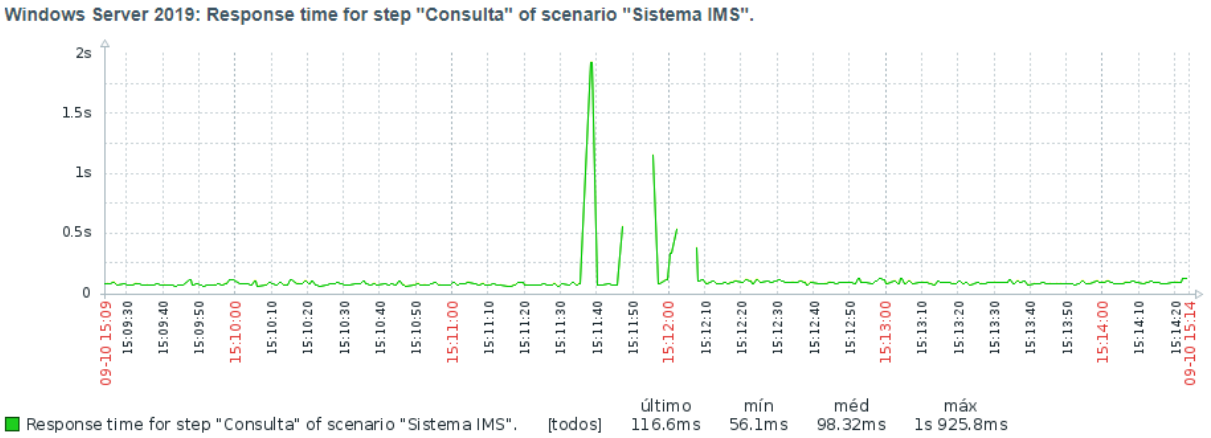
C.2.5 **Tempos de resposta**

Figura 58 – Teste 3 - Tempo de resposta login



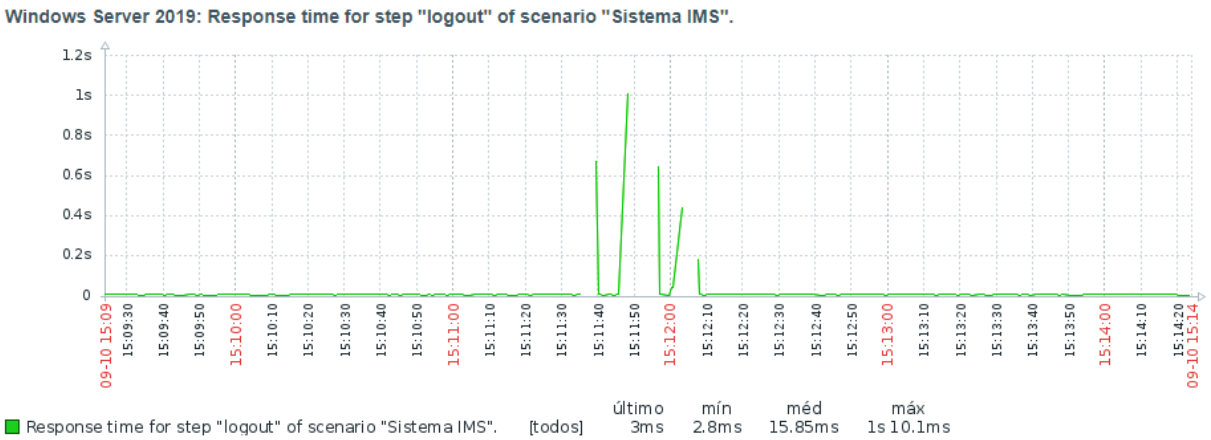
Fonte: Elaborada pelo autor.

Figura 59 – Teste 3 - Tempo de resposta consulta



Fonte: Elaborada pelo autor.

Figura 60 – Teste 3 - Tempo de resposta logout



Fonte: Elaborada pelo autor.

APÊNDICE D – Teste 4 - 400 Usuários

D.1 Dados Apache JMeter

Figura 61 – Teste 4 - Dados JMeter

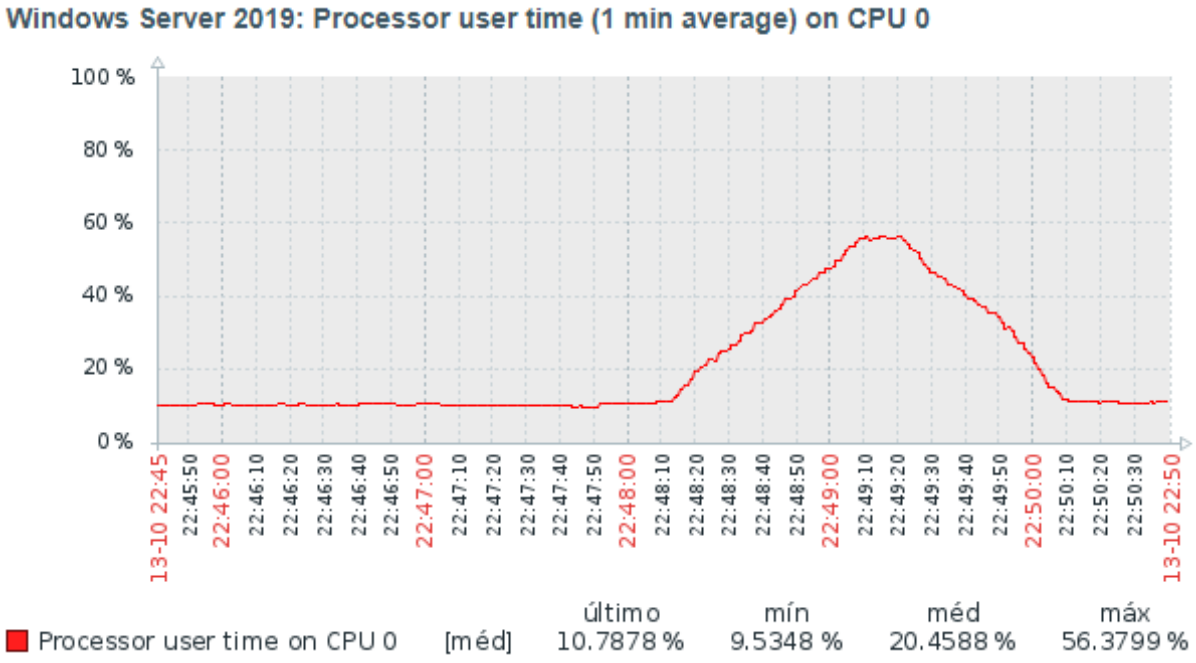
Rótulo	# Amostras	Média	Mín.	Máx.	Desvio Padrão	% de Erro	Vazão	KB/s	Sent KB/sec	Média de Bytes
login	400	663	29	1557	408,67	0,00%	34,8/sec	1469,15	67,10	43199,0
InserirRegistro	400	15614	12342	24143	2858,57	0,00%	9,9/sec	268,54	16,99	27710,0
ConsultarRegistr...	400	7279	129	13649	3197,85	0,00%	15,6/sec	6767,26	7,58	444469,0
logout	400	6172	150	17094	3672,93	0,00%	15,8/sec	49,08	22,94	3182,0
TOTAL	1600	7432	29	24143	6050,46	0,00%	26,6/sec	3365,61	37,07	129640,0

Fonte: Elaborada pelo autor.

D.2 **Dados Zabbix**

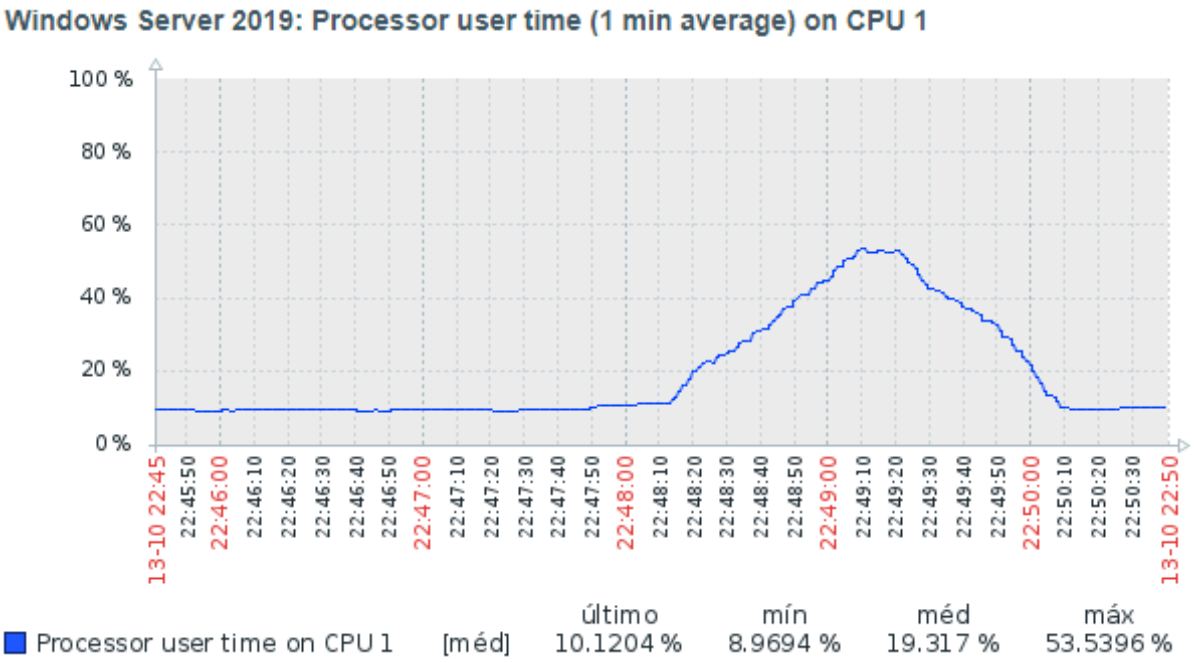
D.2.1 **CPU**

Figura 62 – Teste 4 - Percentual de utilização da CPU 0



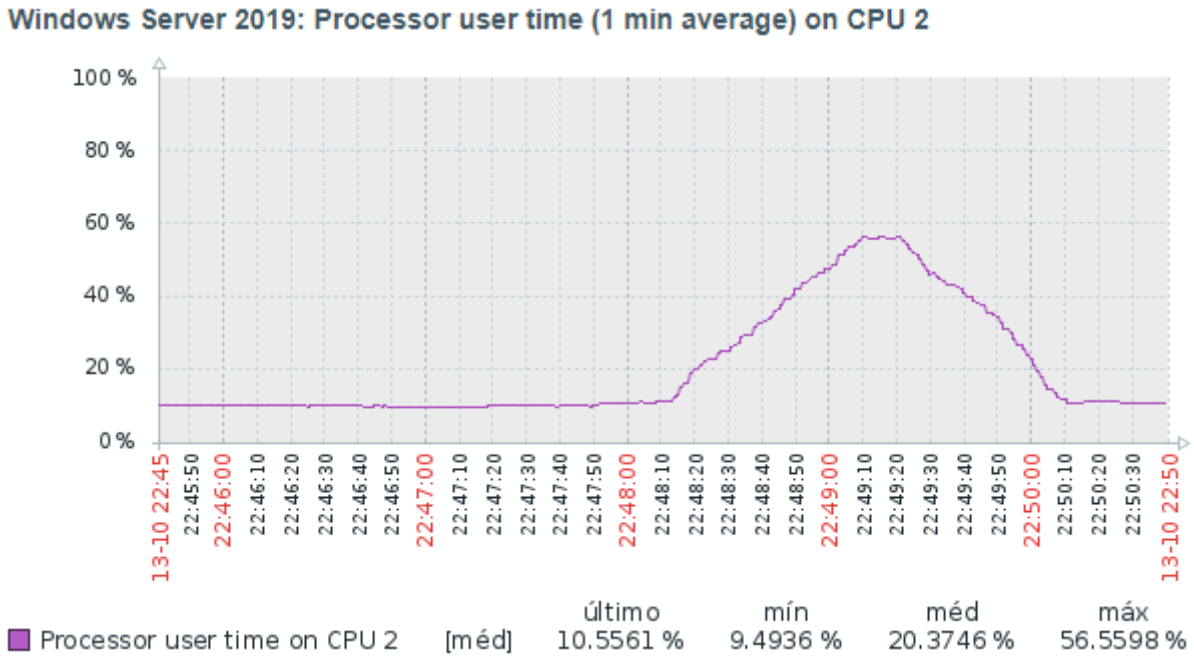
Fonte: Elaborada pelo autor.

Figura 63 – Teste 4 - Percentual de utilização da CPU 1



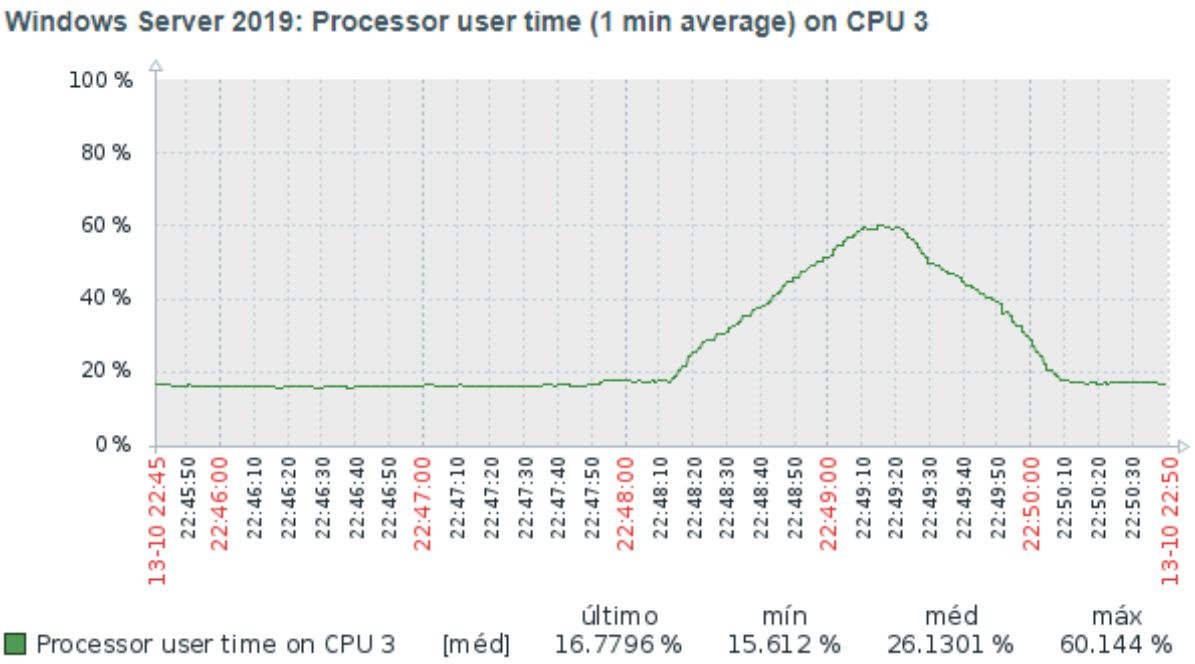
Fonte: Elaborada pelo autor.

Figura 64 – Teste 4 - Percentual de utilização da CPU 2



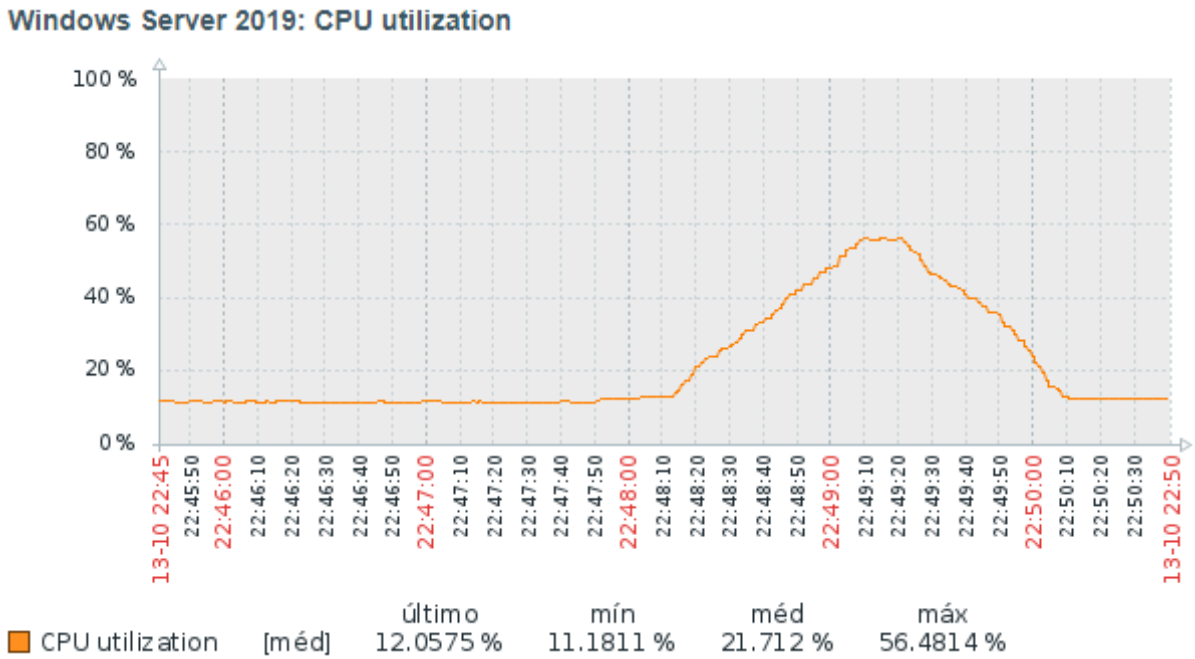
Fonte: Elaborada pelo autor.

Figura 65 – Teste 4 - Percentual de utilização da CPU 3



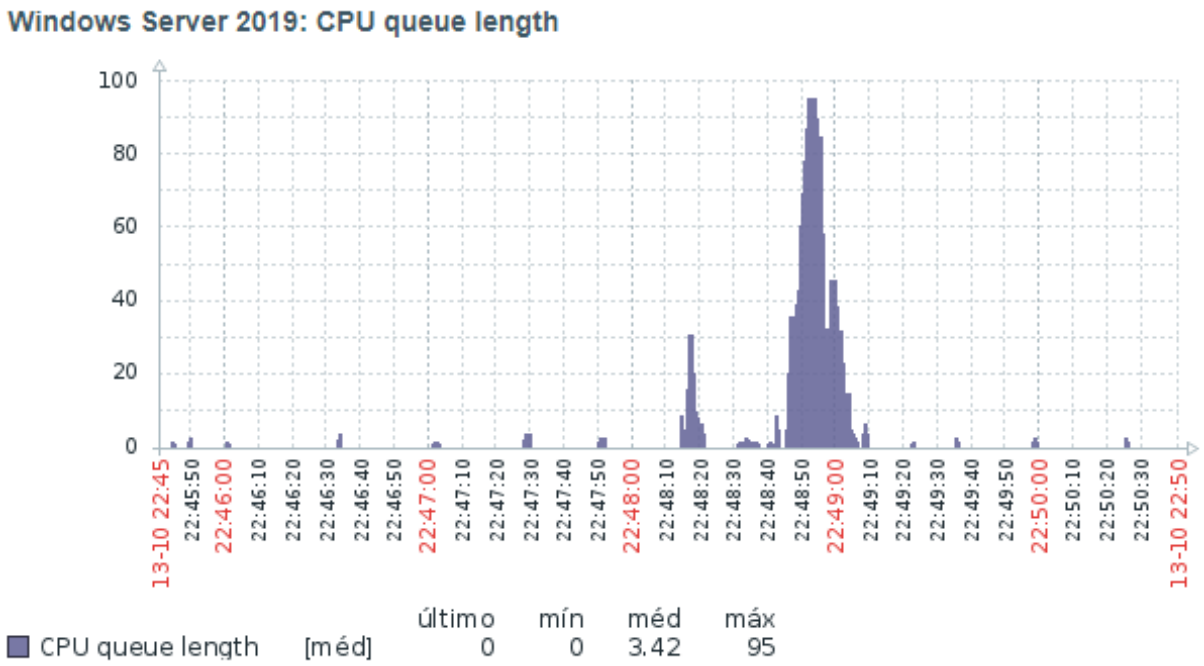
Fonte: Elaborada pelo autor.

Figura 66 – Teste 4 - Percentual de utilização da CPU Total



Fonte: Elaborada pelo autor.

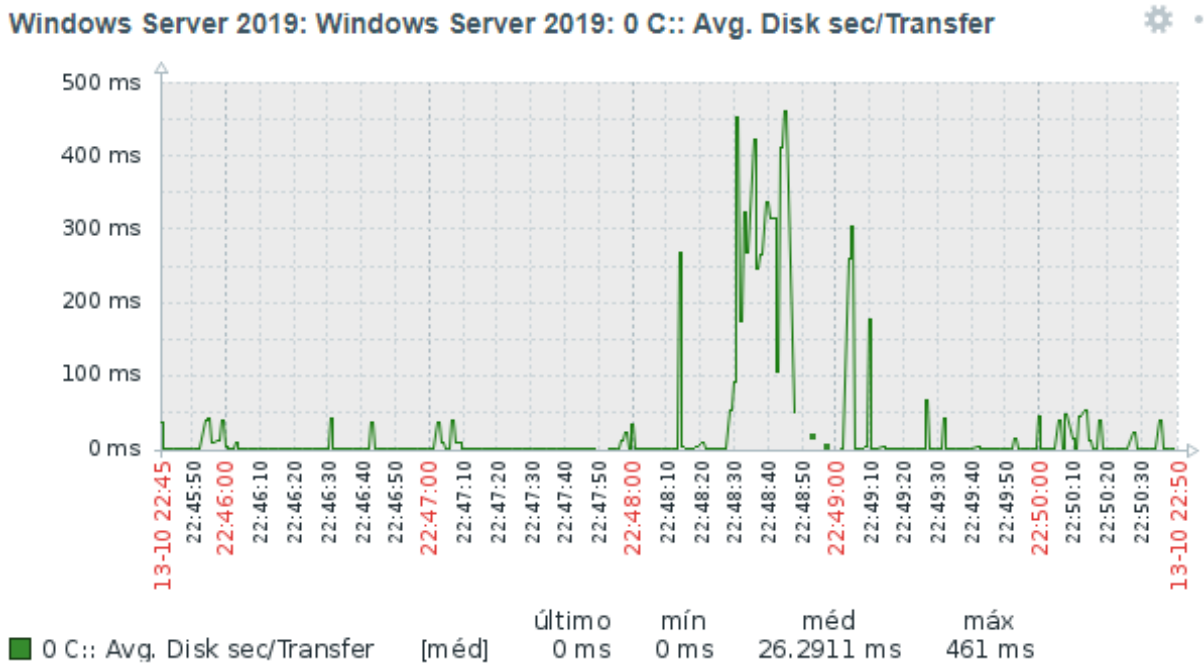
Figura 67 – Teste 4 - Fila de processo na CPU



Fonte: Elaborada pelo autor.

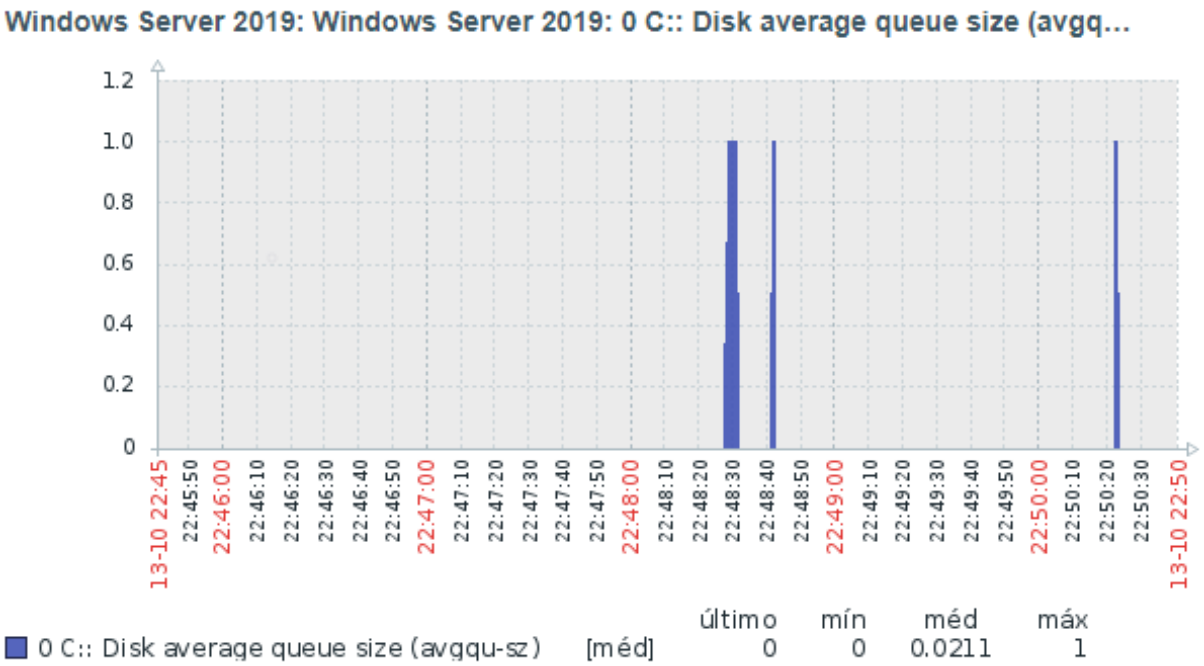
D.2.1.1 **Disco Rígido**

Figura 68 – Teste 4 - Tempo médio de leitura e escrita em disco



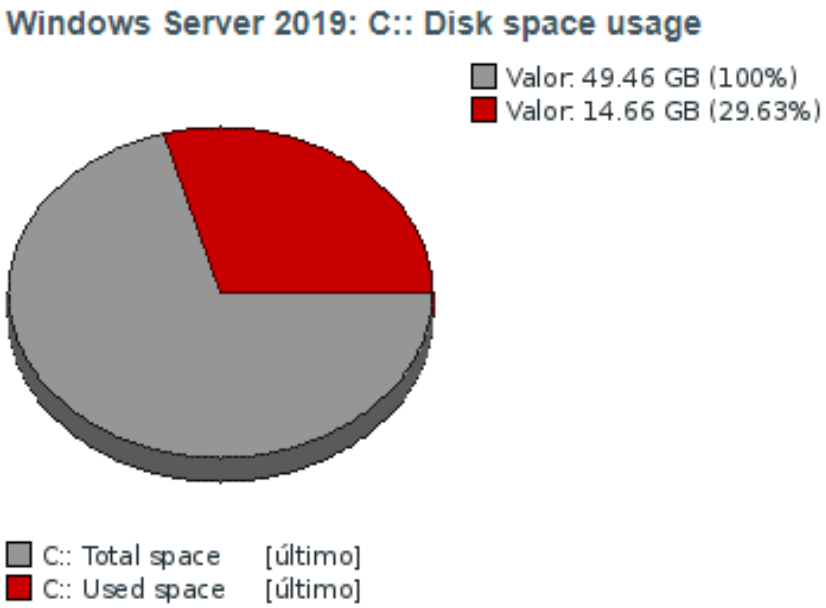
Fonte: Elaborada pelo autor.

Figura 69 – Teste 4 - Média da fila em disco



Fonte: Elaborada pelo autor.

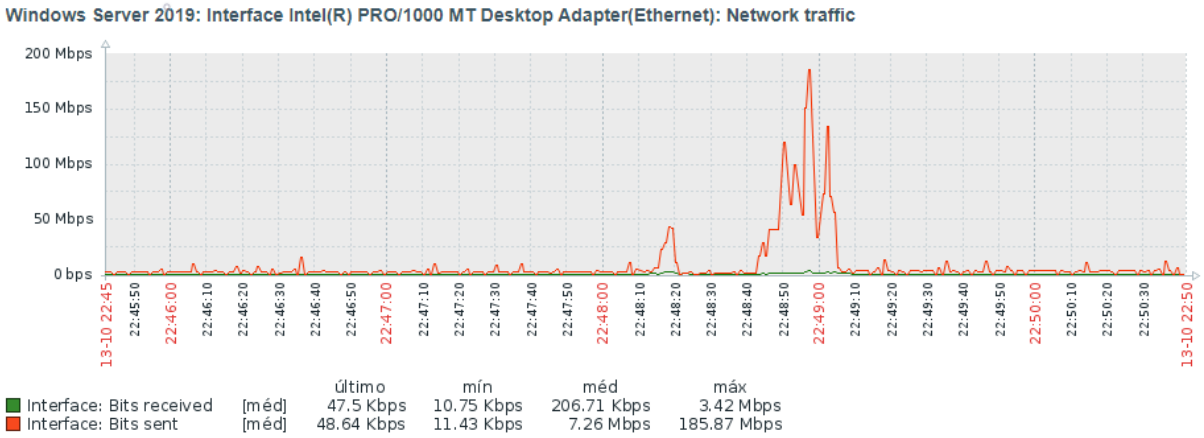
Figura 70 – Teste 4 - Espaço livre em disco



Fonte: Elaborada pelo autor.

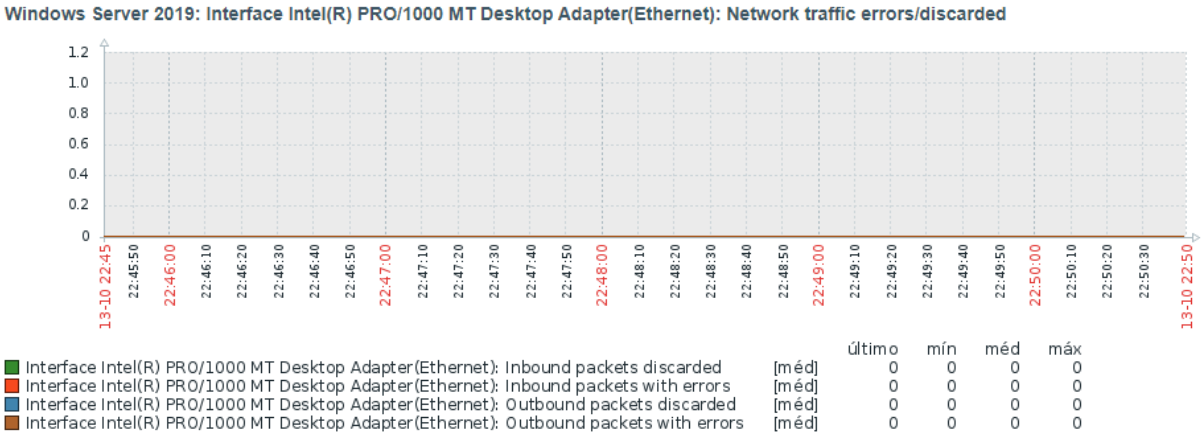
D.2.2 **Placa de rede**

Figura 71 – Teste 4 - Tráfego de rede



Fonte: Elaborada pelo autor.

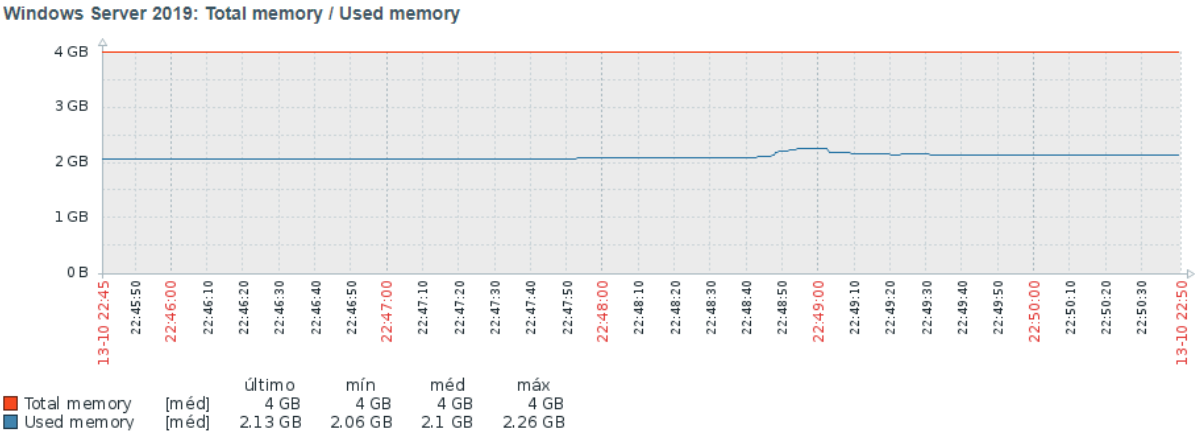
Figura 72 – Teste 4 - Descarte/Erros na Entrada/Saída de pacotes



Fonte: Elaborada pelo autor.

D.2.3 **Memória RAM**

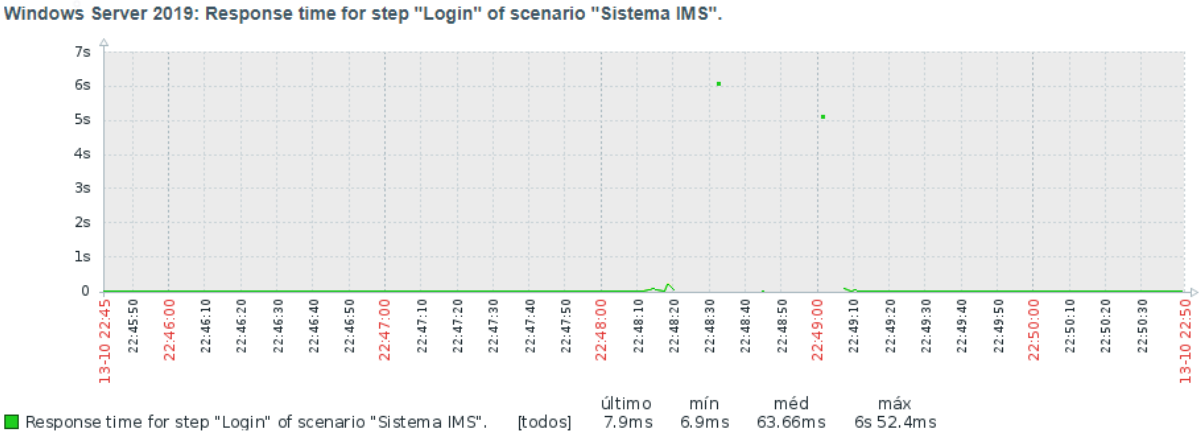
Figura 73 – Teste 4 - Total de memória e memória utilizada



Fonte: Elaborada pelo autor.

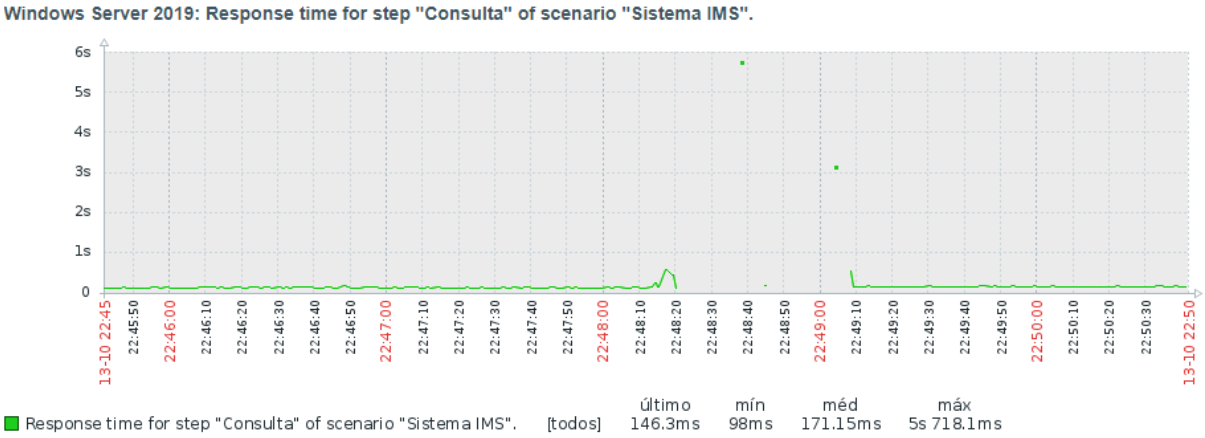
D.2.4 **Tempos de resposta**

Figura 74 – Teste 4 - Tempo de resposta login



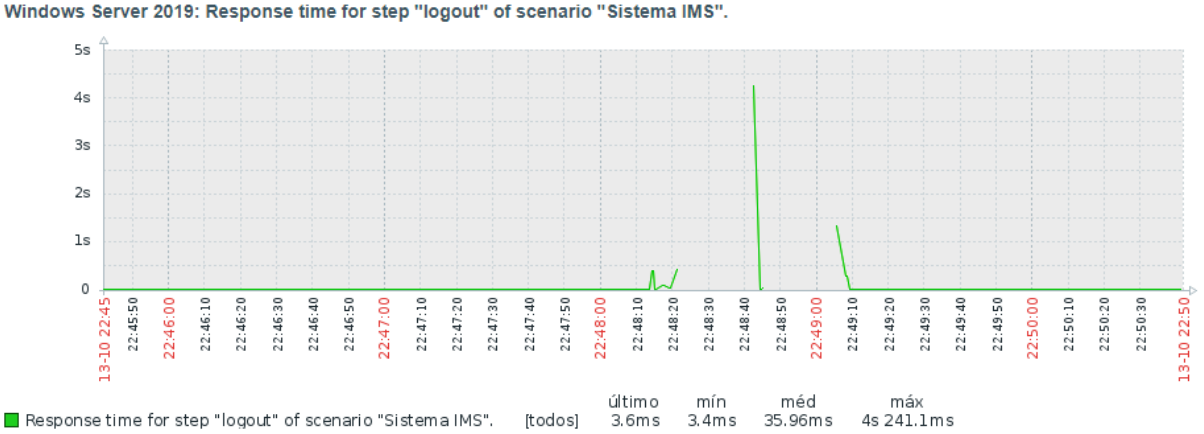
Fonte: Elaborada pelo autor.

Figura 75 – Teste 4 - Tempo de resposta consulta



Fonte: Elaborada pelo autor.

Figura 76 – Teste 4 - Tempo de resposta logout



Fonte: Elaborada pelo autor.