UNIVERSIDADE FEDERAL DE SANTA CATARINA CAMPUS ARARANGUÁ
CENTRO DE CIÊNCIAS, TECNOLOGIAS E SAÚDE
CURSO DE TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO

Júlio César Martins Biz

Monitoramento de redes de computadores via API em sistemas operacionais RouterOS

Araranguá
2023

Júlio César Martins Biz

Monitoramento de redes de computadores via API em sistemas operacionais RouterOS

Trabalho Conclusão do Curso de Graduação em Tecnologias da Informação e Comunicação do Centro de Ciências, Tecnologias e Saúde da Universidade Federal de Santa Catarina como requisito para a obtenção do título de Bacharel em Tecnologias da Informação e Comunicação. Orientador: Giovani Mendonça Lunardi.

Araranguá

2023

Ficha de Identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Martins Biz, Júlio César
Monitoramento de redes de computadores via API em sistemas operacionais RouterOS / Júlio César Martins Biz ; orientador, Giovani Mendonça Lunardi, 2023.
61 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Araranguá, Graduação em Tecnologias da Informação e Comunicação, Araranguá, 2023.

Inclui referências.

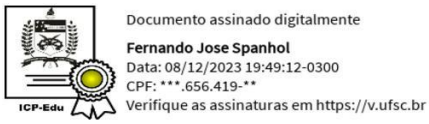
1. Tecnologias da Informação e Comunicação. 2. Monitoramento de redes de computadores. 3. RouterOS. 4. API. 5. MikroTik. I. Mendonça Lunardi, Giovani. II. Universidade Federal de Santa Catarina. Graduação em Tecnologias da Informação e Comunicação. III. Título.

Júlio César Martins Biz

Monitoramento de redes de computadores via API em sistemas operacionais RouterOS

Este Trabalho Conclusão de Curso foi julgado adequado para obtenção do Título de “Bacharel em Tecnologias da Informação e Comunicação” e aprovado em sua forma final pelo Curso de Tecnologias da Informação e Comunicação

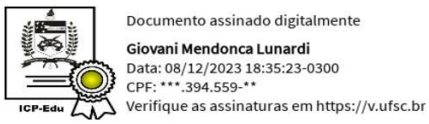
Araranguá, 08 de dezembro de 2023.



Prof. Fernando José Spanhol, Dr.

Coordenador do Curso

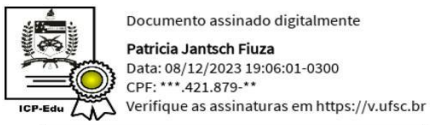
Banca Examinadora:



Prof. Giovani Mendonça Lunardi, Dr.

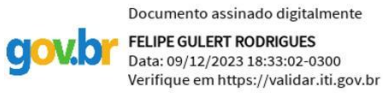
Orientador

Universidade UFSC



Prof.ª Patrícia J. Fiuza, Dra.

Universidade UFSC



Prof. Felipe Gulert

Rodrigues Universidade

UNISACT

Este trabalho é dedicado aos meus pais, irmão, minha noiva e professores, que de alguma forma contribuíram para que eu pudesse chegar até aqui.

AGRADECIMENTOS

Dedico este trabalho, primeiramente a Deus, pois fez com que meus objetivos fossem alcançados, durante todos os meus anos de estudos. Depois, aos meus pais Eládio Biz e Rita de Cássia Martins Biz, por nunca terem medido esforços para me proporcionar um ensino de qualidade. Ao meu irmão João Vitor, pelo companheirismo e principalmente a minha noiva, companheira e amiga, Letícia Custódio Benedet, que com sua cumplicidade e apoio incondicional, esteve em todos os momentos difíceis e compreendeu a minha ausência, enquanto me dedicava à realização deste trabalho. E ao meu orientador, Giovani Mendonça Lunardi, que conduziu o trabalho com paciência e dedicação, sempre disponível a compartilhar todo o seu vasto conhecimento, pelos todos os conselhos, pela ajuda e pela paciência com a qual guiaram o meu aprendizado.

A dúvida é o princípio da sabedoria. (Aristóteles)

RESUMO

Pode-se afirmar que as redes de computadores são essenciais em nossa sociedade globalizada, principalmente em ambientes corporativos, onde a instabilidade da conexão pode gerar prejuízos financeiros. Diante disso, corrobora a importância acompanhar o desempenho e qualidade da rede de internet na qual estes equipamentos estão conectados. Desta forma, este trabalho tem como intuito demostrar o estudo de caso e os benefícios do desenvolvimento de um software na linguagem de programação Python, que realiza o monitoramento de roteadores que utilizam o sistema operacional RouterOS via conexão API (Interfaces de programação de aplicativos). Além disso, essa ferramenta tem por objetivo auxiliar nos desafios enfrentados por profissionais de TI (Tecnologia da Informação), monitorando e coletando informações dos equipamentos. Tais informações disponibilizam dados essenciais para formar juízo sobre um problema, facilitando o monitoramento das oscilações e a disponibilidade da rede, e assim antecipar problemas que possam impactar na infraestrutura.

Palavras-chave: Redes; Internet; RouterOS; API; Python; Monitoramento

ABSTRACT

It can be said that computer networks are essential in our globalized society, especially in corporate environments, where unstable connections can cause financial losses. It is therefore important to monitor the performance and quality of the internet network to which these devices are connected. This paper aims to demonstrate the case study and benefits of developing software in the Python programming language that monitors routers using the RouterOS operating system via API (Application Programming Interfaces) connections. In addition, this tool aims to help with the challenges faced by IT (Information Technology) professionals by monitoring and collecting information from the equipment. This information provides essential data to make a judgment about a problem, making it easier to monitor oscillations and network availability, and thus anticipate problems that could impact the infrastructure.

Keywords: Networks; Internet; RouterOS; API; Python; Monitoring

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

API	*Applications Programming Interface*
CDP	*Cisco Discovery Protocol*
CPU	Unidade Central de Processamento
DNS	*Domain Name System*
DRP	Plano de recuperação de desastres
FCAPS	*Fault, configuration, accounting, performance and security*
HTML	*Hypertext Markup Language*
HTTP	*Hypertext Transfer Protocol*
IoT	*Internet of Things* – Internet das Coisas
ISSO	*International Organization for Standardization*
JSON	*JavaScript Object Notation*
LLDP	*Link Layer Discovery Protocol*
MIB	*Management Information Base*
MNDP	*MikroTik Neighbor Discovery protocol*
OSI	Sistema Aberto de Comunicação
QoS	*Quality of Service*
RAM	*Random Access Memory*
REST	*Representational State of Transfer*
TI	Tecnologia da Informação
TIC	Tecnologia da Informação e Comunicação
VLAN	*Virtual Local Area Network*
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

As redes de computadores desempenham um papel de suma importância à sociedade, fornecendo ferramentas de comunicação que vão desde atividades pessoais até o setor empresarial, proporcionando o desenvolvimento da sociedade. Castells (2003) enfatiza que a Internet é um instrumento fundamental para o desenvolvimento e a globalização.

Além disso, é inegável a importância das redes de computadores e como elas atuam na vida das pessoas mesmo que de forma implícita, pois influencia a maneira que as pessoas se comunicam, seja pelas mídias sociais, trocas de e-mails, videoconferências, mensageiros instantâneos, ou quaisquer outros meios tecnológicos.

Segundo Macedo *et al.* (2018), no mundo corporativo as redes de computadores são empregadas para potencializar diversas atividades, como por exemplo, conectar grandes corporações e suas filiais, ligar sistemas rodando em servidores na nuvem a seus usuários, compra e venda de produtos *on-line* possibilitando a exploração de novas formas de negócios.

> As comunicações de dados e as redes estão mudando a maneira pela qual fazemos negócios e o modo como vivemos. As decisões no mundo dos negócios têm de ser tomadas de forma cada vez mais rápida e aqueles que o fazem precisam obter acesso imediato a informações precisas. (Forouzan, 2006, p. 3)

Diante dessa responsabilidade atribuída às redes de computadores, principalmente em ambientes corporativos, pode-se ratificar que a infraestrutura de TI deve fornecer ferramentas de monitoramento e gerenciamento desses seus equipamentos, a fim de prover uma rede confiável, com alta disponibilidade e produtiva para seus usuários (Souza, 2017).

Além da segurança na operação, o monitoramento de rede também permite a identificação de componentes que estão sendo subutilizados permitindo assim reduzir custo com equipamentos. Isso pode envolver redistribuir a carga de trabalho, consolidar servidores ou até mesmo desativar dispositivos que não estão sendo plenamente aproveitados.

Para Stallings (2005), o gerenciamento de redes é um conjunto de ferramentas utilizadas no monitoramento e controle de rede, devendo conter uma interface para o administrador gerenciar os eventos detectados como, por exemplo, histórico e alertas de falhas, indisponibilidade de servidores e instabilidade na rede. Desta maneira, é possível agir sobre o problema de forma mais rápida e eficiente.

Além disso, Forouzan (2007, p. 876) enfatiza que o gerenciamento do desempenho tem o objetivo de monitorar e controlar a rede para garantir o funcionamento de forma mais eficiente possível.

Assim, o gerenciamento de redes tem como objetivo principal monitorar a eficiência da rede e seus dispositivos, utilizando processos e softwares destinados ao monitoramento e controle de uma rede de computadores, a fim de fornecer ao administrador da rede informações sobre os equipamentos conectados.

Portanto, com o resultado dessas informações o profissional de T.I. pode criar estratégias visando a aumentar à confiabilidade da rede, para que, com base nessas informações, possam antecipar problemas que venham a desestabilizar a infraestrutura.

1.1 PROBLEMÁTICA E JUSTIVICATIVA

O gerenciamento de desempenho da infraestrutura rede no ambiente corporativo é fundamental, para que administradores de TI (Tecnologia da Informação) e equipes de segurança possam identificar eventos e anomalias na rede como, por exemplo, a ausência de banda disponível ou latência excessiva. A fim de evitar falhas que possam gerar grandes prejuízos financeiros, perda de clientes e informações do banco de dados.

Segundo Lyra (2008) a segurança da informação é obtida com o uso adequado de um conjunto de controles, como políticas, processos, procedimentos, estruturas organizacionais e funções de *software* e *hardware*. Esses controles, além de implementados, precisam ser estabelecidos, monitorados e analisados criticamente.

De acordo com Dantas (2011, p. 117) “um sistema de segurança compõe todo um arcabouço de políticas, procedimentos, recursos humanos, tecnologia de suporte

e infraestrutura necessários ao funcionamento das atividades voltadas para a segurança de uma organização”

Portanto, para solucionar ou evitar problemas na infraestrutura de redes, os incidentes e alertas causados pelos equipamentos monitorados devem ser analisados e utilizados para acionar ações corretivas ou até preventivas de acordo com as políticas estabelecidas em cada corporação. Por exemplo, ao ser detectado uma utilização da CPU ou tráfego de rede ou quedas de um *link* de Internet fora dos parâmetros aceitos pela equipe de TI, o sistema deve gerar alertas para que a administração tenha ciência do ocorrido.

> “Gerenciamento de rede inclui a disponibilização, a integração e a coordenação de elementos de hardware, software e humanos, para monitorar, testar, consultar, configurar, analisar, avaliar e controlar os recursos da rede e de elementos, para satisfazer às exigências operacionais, de desempenho e de qualidade de serviço em tempo real a um custo razoável” (Kurose; Ross, 2010, p. 553)

Desta forma, o presente trabalho tem como objetivo apresentar a importância do gerenciamento e monitoramento em roteadores com sistema operacional RouterOS com a finalidade de melhorar o controle sobre a infraestrutura da rede no ambiente corporativo analisado.

Para esse estudo, foi desenvolvido um *software* nomeado DataRouter, com intuito de obter um sistema funcional e eficiente de monitoramento, capaz de fornecer uma visão abrangente e detalhada do desempenho da rede analisada. Além disso, os dados gerados pelo monitoramento fornecem um histórico de falhas e desempenho que podem ser utilizados para prever futuros incidentes.

Este projeto representa não apenas um desafio acadêmico, mas uma oportunidade para buscar soluções de problemas reais e aprofundar os conhecimentos na área de redes de computadores, permitindo explorar soluções para a evolução contínua desse campo crucial da tecnologia.

Acredita-se que a utilização dessa abordagem trará benefícios significativos, para solucionar os problemas de falhas na rede como a redução de tempo de inatividade da rede, prontidão no atendimento, otimização do uso dos recursos do roteador e a melhoria da experiência dos usuários.

Esses dados também evidenciam condições anormais, que devem ser corrigidas num cronograma, com base na criticidade do equipamento e horários disponíveis para intervenção a fim de evitar falhas. Specialski (1999, p. 3) descreve uma falha como “uma condição anormal cuja recuperação exige ação de gerenciamento. Uma falha normalmente é causada por operações incorretas ou um número excessivo de erros”.

Pergunta da pesquisa: Como melhorar a gestão de redes de computadores com o desenvolvimento de ferramentas e *softwares* que disponibilizem informações ao administrador de uma infraestrutura corporativa, utilizando as melhores práticas de gerência de redes?

1.2 OBJETIVOS

Neste subcapítulo são descritos de forma explícita os objetivos deste trabalho, sendo eles o objetivo geral e os objetivos específicos, respectivamente.

1.2.1 Proativa

O gerenciamento de redes proativa busca antecipar e prevenir problemas na infraestrutura de rede. Essa metodologia procura identificar e resolver possíveis falhas antes que elas afetem, negativamente, o desempenho e a disponibilidade da rede (Souza, 2017 p. 16).

Segundo Souza (2017, p. 17), para garantir o bom funcionamento da rede do gerenciamento proativo é essencial identificar potenciais problemas, monitorar constantemente os riscos, planejar e analisar ameaças, além de manter um gerenciamento contínuo para lidar com mudanças internas e externas.

1.2.2 Objetivo Geral

Desenvolver um *software* para realizar o monitoramento de roteadores que utilizam o sistema operacional RouterOS via API visando a fornecer alertas para os administradores de rede.

1.2.3 Objetivos Específicos

Por meio dos objetivos específicos, será definido o caminho para chegar ao objetivo geral:

- Coletar e analisar métricas de uso de *hardware* e *status* de conexões via API RouterOS;
- Gerenciar configurações dos equipamentos Mikrotik cadastrados no *software*;
- Analisar os cenários para realizar a detecção de eventos e enviar alertas em tempo real;
- Apresentar os benefícios do monitoramento pelo *software* desenvolvido.

1.3 ADERÊNCIA AO CURSO DE TIC

Apesar ser possível encontrar inúmeras ferramentas no mercado prontas e maduras para a criticidade de um ambiente corporativo como Paessler PRTG Network Monitor, Zabbix, Graylog, Grafana, este trabalho teve o intuito de reunir diversos conhecimentos adquiridos ao logo da jornada acadêmica no Bacharelado em Tecnologias da Informação e Comunicação – TIC, sendo eles o desenvolvimento de softwares, modelagem de banco de dados, gerenciamento redes de computadores e gestão de ativos.

> "É um curso de computação aplicada, que visa formar profissionais capazes de solucionar problemas que envolvem a utilização de Tecnologias da Informação e Comunicação (TIC) em organizações. As soluções poderão ter uma ênfase em sistemas de informação, negócios ou aspectos ligados à educação e cultura" (UFSC).

O quadro 1 apresenta os trabalhos identificados nas bases de dados de Trabalhos de Conclusão de Curso (TCCs) da UFSC. Com base nesses três trabalhos, percebe-se que apresentam a temática monitoramento de redes nas organizações, mas não abordam sobre a base para o desenvolvimento de uma ferramenta de monitoramento.

Quadro 1 - Aderência aos trabalhos de conclusão do curso de TIC.

Título	Autor	Ano
Análise de Ferramentas para Segurança de Redes	Ancelmo Boteon	2007
Monitoramento de Servidores com Scripts	André Felipe Durieux	2012
Monitoramento de Ativos de Rede Utilizando Softwares Open-Source	Rainer Testa Medrado	2018

Fonte: Elaborado pelo autor

1.4 ESTRUTURA DO TEXTO

O conteúdo desta monografia foi estruturado em cinco capítulos a fim de facilitar o entendimento acerca do que foi desenvolvido, sendo eles:

- Capítulo um, apresenta a introdução e contextualiza o trabalho, além de elencar os tópicos a respeito dos objetivos, da problemática e da justificativa;

- No capítulo dois, será abordado o referencial teórico, com a finalidade de justificar as escolhas relacionadas às tecnologias utilizadas;

- No capítulo três, apresenta a metodologia utilizada durante o desenvolvimento do trabalho e informações sobre a cenário analisado;

- No capítulo quatro é apresentado as tecnologias utilizadas na pesquisa para obtenção dos resultados e os indicadores desenvolvidos, bem como os métodos de desenvolvimento que serão abordados no seguinte capítulo;

- No capítulo cinco, são apresentados os resultados e discussões sobre os resultados do trabalho;

- Capítulo seis apresenta as conclusões que o trabalho alcançou, dificuldades encontradas, além de ações e trabalhos futuros recomendados para a replicação da metodologia de monitoramento de redes.

2 FUNDAMENTAÇÃO TEÓRICA

O presente capítulo tem por objetivo apresentar a base teórica dos estudos relacionados com este trabalho. Para tanto, serão expostos os tipos de gerenciamento de redes e suas peculiaridades, modelo de gerência de redes OSI (*Open Systems Interconnection*) tratando individualmente de cada uma das cinco áreas de monitoramento.

Em seguida, apresentam-se as tecnologias de integração utilizadas no projeto, sendo elas o protocolo TCP/IP *(Transmission Control Protocol/Internet Protocol*) e HTTP (*Hypertext Transfer Protocol*), conexões por *sockets* e API (Interface de Programação de Aplicação), discutindo sua importância e aplicabilidade no contexto do trabalho em questão

2.1 TIPOS DE GERÊNCIA DE REDES

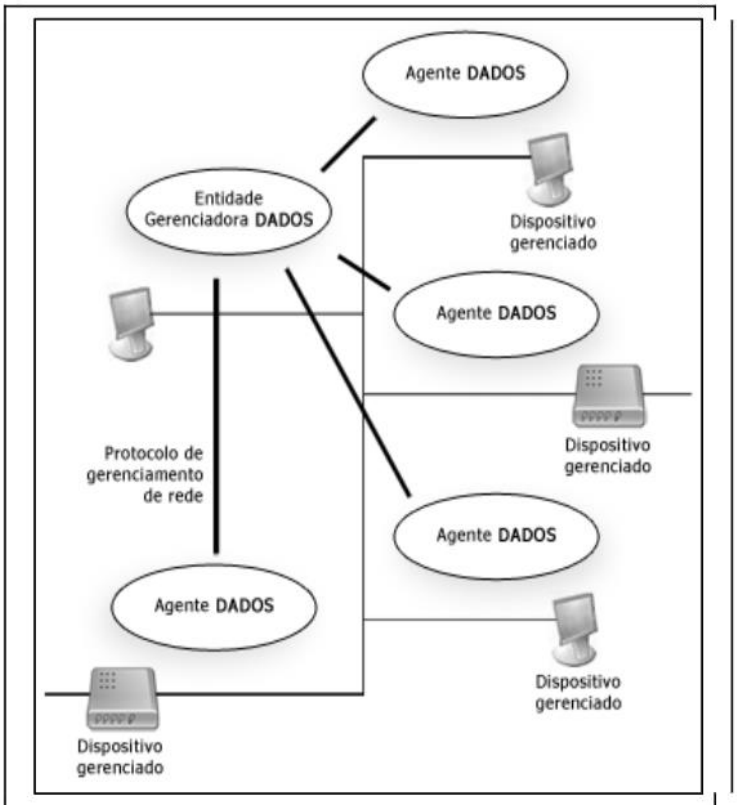
Os componentes de uma arquitetura de gerência de rede têm sua própria terminologia e definições para os componentes do monitoramento, sendo eles: entidade gerenciadora, dispositivo gerenciado e protocolo de gerenciamento (Sousa, 2017, n.p).

A primeira entidade gerenciadora é a aplicação responsável pela obtenção e o envio de informações de gerenciamento, mediante a comunicação com um ou mais dispositivos gerenciados. O dispositivo gerente fica responsável pelo monitoramento, relatórios e decisões na ocorrência de problemas.

Além disso, o dispositivo gerenciado pode ser definido como um ativo de rede que integra um conjunto de objetos gerenciáveis constituídos por componentes de hardware e software. O agente fica responsável pelas funções de envio e alteração das informações.

E também, o protocolo de gerenciamento é responsável pela comunicação entre a entidade gerenciadora e o agente de gerenciamento. Desta forma, permitindo que a entidade gerenciadora colete dados dos objetos gerenciáveis e se necessário, execute ações sobre eles mediante seus agentes

Figura 1 - Principais componentes de uma arquitetura de gerenciamento



Fonte: (Kurose; Ross, 2006)

As informações resultantes do monitoramento do dispositivo gerenciado são organizadas em uma base de dados denominada MIB (*Management Information Base*), que pode ser acessada e modificada pela entidade gerenciadora (Battisti, 2007, p. 20).

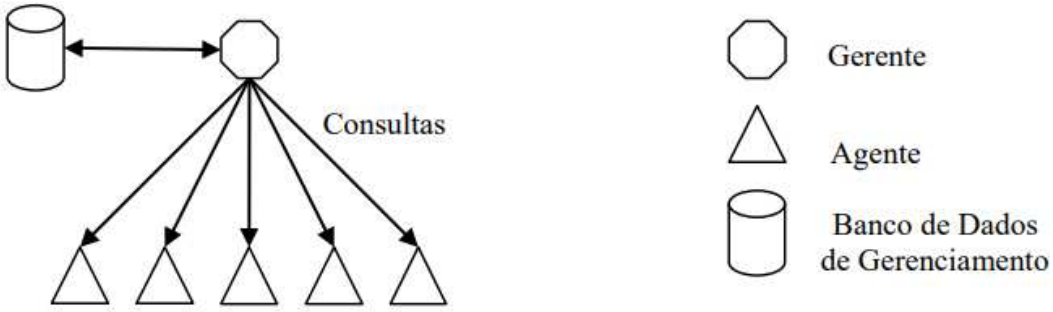
Essa base de dados pode estar centralizada ou distribuída, conforme o tipo de estrutura de gerência de dados, nesse subcapítulo são descritos de forma explícita os objetivos benefícios e desvantagens de cada uma (Sousa, 2017, p. 15-18).

2.1.1 Centralizada

Esse modelo de gerência de redes centraliza o modelo em que todas as informações são enviadas pelos softwares agentes instalados em cada dispositivo gerenciado, esses dados são centralizados em um servidor que é responsável pelo monitoramento e gerenciamento dos dispositivos (Battisti, 2007, p. 20).

Sua simplicidade garante consistência nas práticas de gestão, eficiência operacional e uma visão abrangente da infraestrutura de rede. Porém, devido a sua centralização, isso gera um tráfego intenso e por ter um único elemento de monitoramento, em caso de falha todo o gerenciamento para de funcionar.

Figura 2 - Arquitetura de gerenciamento centralizado

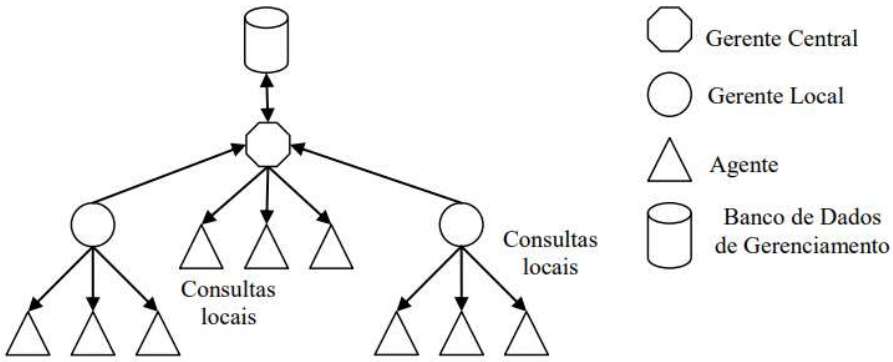


Fonte: (Battisti 2007, p. 20)

2.1.2 Descentralizada

Uma das vantagens deste modelo é a distribuição do tráfego e processamento em vários “nós”. Sendo assim, as verificações serão distribuídas em servidores subjacentes na infraestrutura.

Figura 3 - Arquitetura de gerenciamento descentralizado



Fonte: (Battisti 2007, p. 20)

2.1.3 Reativa

O modelo de gerenciamento de redes reativo é uma abordagem em que as ações são tomadas em resposta a eventos ou falhas identificados na rede (Forouzan, 2007 p. 875). Sendo assim, os dispositivos que estão sendo monitorados são configurados para emitir um alerta ao sistema de gerenciamento central, após detectar um problema na rede.

Assim, uma das limitações no modelo de gerenciamento reativo por depender da detecção de problemas após eles ocorrerem é a possibilidade de haver um tempo de resposta significativo na solução do problema, o que pode resultar em interrupções prolongadas ou perda de dados.

Segundo Forouzan (2007, p. 875) define que o “gerenciamento de falhas reativo” possui quatro atribuições:

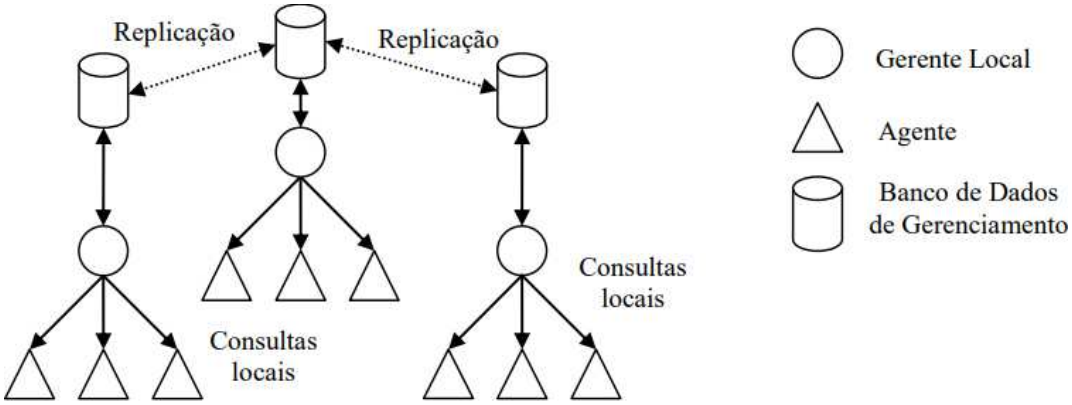
- Detecção - O primeiro passo de um sistema de gerenciamento de falhas reativo é detectar a localização exata da falha;
- Isolamento - Quando ocorre uma falha, o sistema para de funcionar corretamente ou o sistema cria erros excessivos, esse procedimento é importante para diminuir a quantidade de falhas e usuários afetados;
- Correção - O terceiro passo é corrigir a falha. Isso pode envolver a substituição ou reparo de equipamentos;
- Documentação - Uma vez resolvida a falha, é essencial realizar sua documentação. O registro deve conter informações precisas sobre a localização exata da falha, possíveis causas, ações tomadas para corrigi-la, bem como o custo e o tempo envolvidos em cada etapa. A documentação desempenha um papel crucial, como analisar a frequência dos eventos, estimar vida útil dos equipamentos, permitir que outros administradores de rede estejam cientes do incidente.

2.1.4 Distribuída

No modelo de gerenciamento distribuído cada dispositivo gerente coleta e processa as informações da rede e armazena no seu próprio banco de dados. Além disso, os bancos de dados são replicados entre si, proporcionando maior segurança,

pois os dados estão replicados. Desta forma, em caso de falha de um agente, ainda será possível restaurar as informações necessárias (Souza, 2017 p. 17).

Figura 4 - Modelo de Gerenciamento distribuído



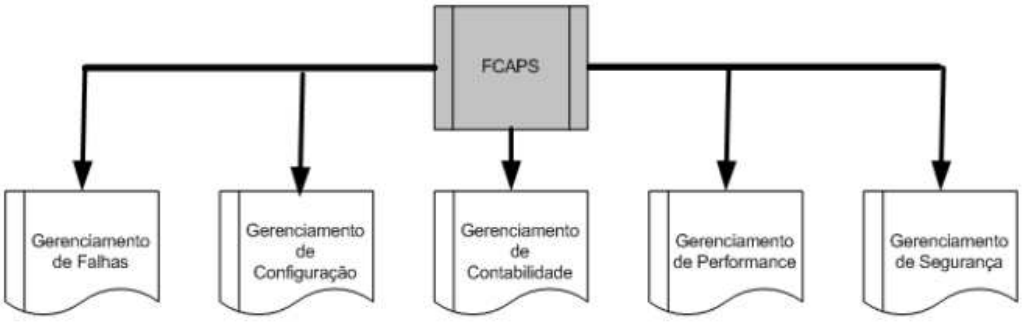
Fonte: (Battisti 2007, p. 21)

2.2 MODELO DE GERÊNCIA DE REDES OSI

O modelo de gerência de redes de computadores OSI (*Open Systems Interconnection*) é uma estrutura de gerenciamento desenvolvida pela ISO (*International Organization for Standardization*). O principal objetivo da instituição foi criar classificações de gestão, nas quais o modelo OSI (*Open Systems Interconnection*) define as tarefas de gerenciamento de rede conhecidas como FCAPS acrônimo *para fault, configuration, accounting, performance e security*.

Segundo Sousa (2017), “Essas cinco áreas visam atender aos requisitos de desempenho, segurança, funcionalidade e tempo de resposta (qualidade de serviço – QoS)”.

Figura 5 - Tarefas de gerenciamento FCAPS



Fonte: (Bianchini 2016, p. 18)

2.2.1 Gerência de Falhas (*Fault*)

O objetivo principal do gerenciamento de falhas é a detecção do problema, isolamento, análise, diagnóstico e a correção, com o objetivo de garantir a operação contínua da rede (Sousa, 2017, p. 19).

Uma falha é uma condição anormal persistente que causa a indisponibilidade de um recurso ou serviço na rede. Por exemplo, a queda de uma interface de rede que resulta na indisponibilidade de um *link* de Internet. Essa indisponibilidade pode ser causada por problemas físicos, como cabos danificados ou dispositivos defeituosos, ou por problemas lógicos, como erros de configuração nos equipamentos de rede.

Segundo Eler (2015, p. 4) o gerenciamento de falhas pode ser definido como a “função de monitorar os estados dos recursos verificando em qual ponto da rede e quando uma falha ou um erro pode ocorrer”.

2.2.2 Gerência de Configuração (*Configuration*)

O gerenciamento de configuração tem como objetivo a monitoração e controle da rede, que engloba a instalação de equipamentos, configuração e atualização. Realizando registros de inventário das configurações e recursos a rede (Sousa, 2017, p. 20).

Essa gerência desempenha um papel crucial, uma vez que diversos problemas de rede surgem como resultado direto de modificações de configurações de sistemas, atualizações de software ou alterações na infraestrutura de *hardware*. Um modelo eficiente de gerenciamento de configuração abrange o controle meticuloso de todas as alterações realizadas no hardware e software da rede.

Para Eler (2015, p. 4), o gerenciamento de configuração “permite manter atualizadas as informações de hardware e software de uma rede, incluindo as informações de configurações de todos os equipamentos”.

2.2.3 Gerência de Contabilização (*Accounting*)

O objetivo da gerência de contabilização é fornecer uma visão precisa e detalhada do consumo de recursos da rede, permitindo um controle efetivo de custos, alocação eficiente de recursos e planejamento de capacidade. Além disso, a contabilização também desempenha um papel importante na detecção de abusos ou violações de políticas, auxiliando na implementação de medidas de segurança e controle de acesso adequadas (Eler, 2015, p. 4).

2.2.4 Gerenciamento de desempenho *(performance)*

A Gerência de Desempenho é uma área fundamental no gerenciamento de redes, que tem como objetivo a medição, monitoramento e otimização do desempenho da rede. Com essas informações, é possível tomar ações proativas para otimizar a eficiência e garantir uma experiência de rede satisfatória para os usuários. (Sousa, 2017 p. 20).

Desta forma, essas informações fornecem dados valiosos para identificar situações problemáticas, como o uso excessivo do *link* de internet, nível de CPU ou Memória RAM de um servidor, que podem resultar em lentidão e problemas de conexão, antes que afetem negativamente os usuários finais.

2.2.5 Gerenciamento de segurança (*security)*

O gerenciamento de segurança desempenha diversas tarefas essenciais para garantir a integridade da rede como por exemplo:

- Manutenção e distribuição de senhas;
- Configurações de controle de acesso;
- Coletar, armazenar e analisar *logs*;
- Geração, distribuição e armazenamento de chaves de criptografia.

Segundo Forouzan (2007, p. 876), o objetivo do gerenciamento de segurança é estabelecer o controle de acesso à rede, seguindo uma política de segurança pré-definida com o objetivo de proteger recursos da rede e informações dos usuários.

2.3 RECURSOS TECNOLÓGICOS DO SOFTWARE

2.3.1 Tecnologias de integração

A interconexão de sistemas se tornou extremamente importante, à medida que as organizações estão cada vez mais procurando por soluções integradas (Silva, 2004). Desta forma, para garantir o sucesso do processo de integração entre as aplicações, é fundamental que as ferramentas de integração sejam adequadas às necessidades específicas (Karl, 1994).

É essencial que as soluções escolhidas sejam capazes de se adaptar aos requisitos técnicos e funcionais das aplicações envolvidas, permitindo uma interação fluida e eficiente entre os sistemas. Ao selecionar ferramentas compatíveis e adequadas, as organizações têm maiores chances de obter êxito na implementação de um processo de integração eficaz (Silva, 2004).

Segundo Bisol (2019) enfatiza que “Em uma organização pode haver inúmeros sistemas, a informação contida em cada um desses sistemas muitas vezes precisa estar compartilhada nos diversos setores da empresa, por isso a necessidade de integração entre sistemas”.

Em última análise, a integração de sistemas simplifica a automação de tarefas, reduzindo erros e tempo gasto em processos manuais. Além disso, a integração de sistemas é essencial para a colaboração entre departamentos e equipes.

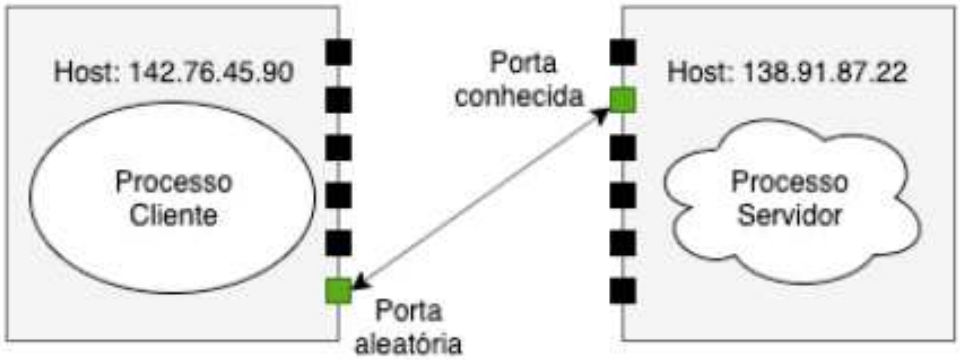
Assim, diante da importância de criar aplicações que garantam uma comunicação contínua e confiável, nas subseções abaixo serão citadas as tecnologias que foram utilizadas na comunicação entre o software DataRouter e o sistema operacional RouterOS.

2.3.2 Conexão socket

Um *socket* é uma interface de comunicação bidirecional entre processos distintos que podem estar na mesma máquina (*Unix Socket*) ou por redes de computadores, utilizando o protocolo TCP/IP ou UDP/IP no modelo cliente-servidor (Maziero, 2008).

Além disso, *Sockes* em redes de computadores utilizam a arquitetura de comunicação cliente-servidor, no qual o processo servidor permanece em um estado de espera constante, aguardando que um processo cliente inicie a comunicação, o qual possui conhecimento do endereço de IP e da porta de acesso ao servidor. Nesse contexto, o cliente toma a iniciativa de estabelecer a conexão, enquanto o servidor atua de forma passiva, pronto para responder às solicitações do cliente (Maziero, 2008).

Figura 6 - Comunicação utilizando *Socket*



Fonte: (Tedesco, 2019)

E também, os *sockets* do tipo TCP que seguem o protocolo de transporte orientado a conexão, estabelecem um canal dedicado de comunicação entre cliente e servidor, assegurando a ordem e a confiabilidade na entrega de pacotes. No entanto, sua abordagem é mais burocrática e lenta quando se trata de lidar com falhas e perda de pacotes, devido aos procedimentos robustos de controle de erro e retransmissões que implementam (Perin, 2022).

Outro modelo de comunicação de *sockets* é UDP que não se preocupam com a garantia de entrega ou recuperação de falhas. Porém, essa característica os torna mais rápidos em comparação ao protocolo TCP. Devido, as características do protocolo são utilizadas em aplicações que toleraram perda de dados (Perin, 2022).

2.3.3 API

A *Application Programming Interface* (API) é um conjunto de diretrizes que define formatos de dados, operações disponíveis e protocolos estabelecidos por uma aplicação permitindo o desenvolvimento de plataformas e serviços se conectem e compartilhem informações de maneira eficiente e segura (Oliveira, 2018).

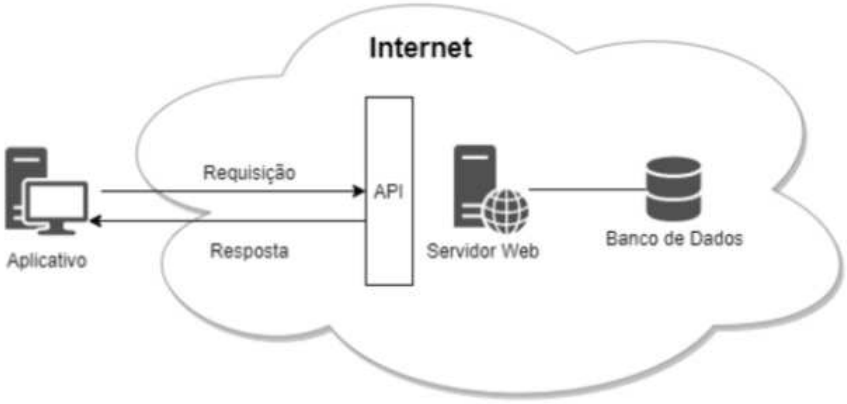
Portanto, as APIs atuam como interfaces entre sistemas distintos e podem ser vistas como contratos, que representam um acordo de quais requisições podem ser feitas e como o software da outra parte responderá (Queiroz; Conceição; Barreto, 2021, p.3).

Segundo Stylos *et al.* (2009) As interfaces de programação de aplicativos (APIs) funcionam como meios pelos quais uma aplicação de software disponibiliza seus serviços ou dados. Elas oferecem acesso a um conjunto predefinido de recursos, permitindo a interação e integração com outras aplicações ou sistemas (*apud* Fernandes, 2019, p. 36). Desta forma, tal protocolo permite que um dispositivo acesse informações de banco de dados, sensores sem ser necessário entregar acesso total ao dispositivo cliente.

Na arquitetura cliente-servidor, os dados são centralizados nos servidores, que geralmente possuem níveis mais elevados de segurança em comparação com a maioria dos clientes. Os servidores têm a capacidade de gerenciar com mais eficácia o acesso aos serviços, assegurando que apenas os clientes com credenciais válidas

tenham permissão para utilizar os serviços disponibilizados, o que fortalece a segurança e o controle de acesso na infraestrutura. (Queiroz; Conceição; Barreto, 2021, p.3).

Figura 7 - Diagrama requisição API – cliente-servidor



Fonte: (Queiroz; Conceição; Barreto, 2021, p.19)

A API do RouterOS usa uma codificação específica para representar os comandos e as respostas trocadas entre o cliente e o roteador. A codificação é semelhante a uma forma de serialização de dados e é conhecida como "codificação de palavras" (Word Encoding). A API do RouterOS codifica e decodifica os dados usando esse formato para garantir que os comandos e respostas sejam transmitidos de maneira eficiente e precisa.

Além disso, a API do sistema operacional RouterOS segue a sintaxe da interface de linha de comando (CLI). Quando o roteador recebe uma frase ela é avaliada e executada, então uma resposta é formada e retornada

A estrutura básica de uma mensagem na codificação de palavras é a seguinte:

Figura 8 - Exemplo uso API do RouterOS em Python

```python
import os
from dotenv import load_dotenv
load_dotenv()

from api import Api

router = Api(address=os.environ["ip_routerboard"],
             user=os.environ["user_router_os"],
             password=os.environ["pw_router_os"],
             port=os.environ["port_routerboard"])


def exemplo_uso_api():

    try:
        response = router.talk('/interface/print')
        response

        print(response)

    except router.LoginError(Exception) as e:
        print(f"Ocorreu um erro na requisição: {e}")
```

Fonte: Elaborado pelo autor

2.3.4 Pyton

A linguagem de programação Python[1] é uma linguagem de código aberto, orientada a objeto, com licença compatível com a *General Public Licence* (GPL), apresentando tipagem dinâmica e forte. Além disso, Python é multiparadigma, oferecendo suporte tanto para programação modular quanto funcional, além da orientação a objetos. Notavelmente, até mesmo os tipos básicos em Python são tratados como objetos, ampliando a flexibilidade e expressividade da linguagem para os desenvolvedores, que têm à disposição diversas abordagens para diferentes estilos de programação (Borges, 2014, pg. 14).

> “O suporte melhorado de Python para bibliotecas (como o pandas e o scikit-learn) o transformou em uma opção popular para tarefas de análise de dados. Em conjunto com a robustez de Python para uma engenharia de software de propósito geral, é uma excelente opção como uma linguagem principal para a construção de aplicações de dados”. (McKinney, 2011, p10)

[1] https://www.python.org/

Desta forma, a escolha da linguagem Python para o desenvolvimento foi devido à ampla gama de estruturas de alto nível, como listas, dicionários, manipulação de data/hora, números complexos, além da extensa coleção de módulos prontos para uso, juntamente com *frameworks* de terceiros que podem ser facilmente incorporados. O que permite aos desenvolvedores e analistas manipular dados de forma eficiente e realizar operações estatísticas complexas.

2.3.5 MongoDB

O MongoDB[2] é um banco de dados não-relacional baseado em documentos, em sua arquitetura esses documentos não apenas contêm os dados, mas também definem sua estrutura composta por pares de chave e valor em um arquivo JSON (MongoDB).

O modelo baseado em documentos oferece flexibilidade à aplicação, uma vez que não se utiliza de tabelas e colunas pré-definidas e que não se prende a um esquema rígido de um banco de dados relacional. Dessa forma, é possível atualizar a estrutura do documento, adicionando-se novos campos que não irão causar nenhuma inconsistência ao banco de dados (Lóscio *et al*., 2011).

Os documentos do MongoDB contêm uma chave única de identificação especial “_id”, que por sua vez é único também dentro da coleção de documentos, que é um agrupamento de vários documentos, ou seja, identifica o documento globalmente dentro da coleção (Hecht; Jablonski, 2011).

Conforme figura 9 se pode observar o equipamento monitorado retornou duas respostas diferentes durante as requisições de teste de PING com o servidor DNS. Como se pode perceber são estruturas de dados diferentes e mesmo assim o banco MongoDB consegue lidar com essas diferenças. Sendo uma responsabilidade da aplicação fazer o tratamento de dados durante as consultas ou inserções.

[2] https://www.mongodb.com/

Figura 9 - Exemplo estrutura de documento banco de dados MongoDB

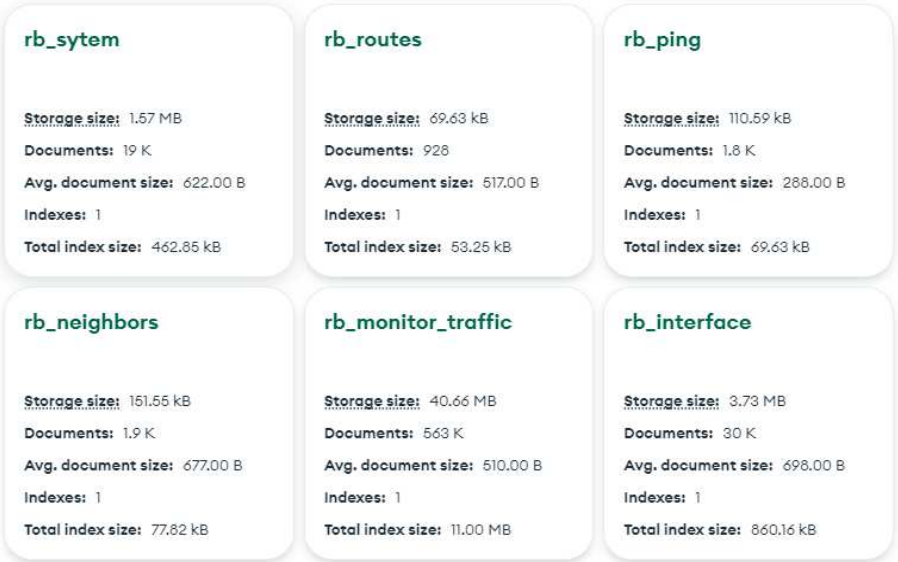

```
_id: ObjectId('6561b181edb7883793efa7cf')
seq: "4"
host: "1.1.1.1"
size: "56"
ttl: "51"
time: "22ms581us"
sent: "5"
received: "5"
packet-loss: "0"
min-rtt: "22ms441us"
avg-rtt: "22ms585us"
max-rtt: "22ms707us"
data: "2023-11-25"
hora: "05:34"
data-hora: "2023-11-25T05:34:09.124545"
```

```
_id: ObjectId('6561b1c1edb7883793efab4a')
seq: "0"
status: "packet rejected"
sent: "1"
received: "0"
packet-loss: "100"
data: "2023-11-25"
hora: "05:35"
data-hora: "2023-11-25T05:35:13.896319"
```


Fonte: Elaborado pelo autor

Coleções são grupos de documentos. No MongoDB, essas estruturas equivalem às tabelas dos bancos de dados relacionais.

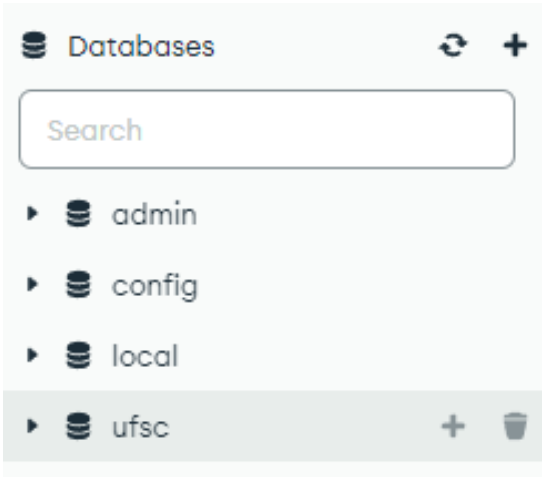
Figura 10 - Coleções do projeto DataRouter



Fonte: Elaborado pelo autor

Os servidores do MongoDB podem armazenar diferentes databases que são contêineres que armazenam diversas coleções conforme exemplo da figura 11.

Figura 11 - Exemplo de Databases em um servidor MongoDB



Fonte: Elaborado pelo autor

2.3.6 Sistema operacional RouterOS

O sistema operacional nomeado RouterOS[3], foi desenvolvido em 1997 pela empresa Mikrotik[4] com sede na Letônia, para uso em seus equipamentos, como roteadores, *switches*, *access-points* e antenas de rádio.

Baseado no kernel Linux sendo um sistema licenciado, *stand-alone* com diversas funcionalidades em redes de computadores roteamento, como *Firewall*, AP (*access point*), Servidor VPN (*Virtual Private Network*), Servidor DHCP (*Dynamic Host Configuration Protocol*), servidor DNS (*Domain Name System*), entre outros. Podendo ser instalado diretamente no *hardware* da mikrotik denominado RouterBOARD[5], além de possuir versões para processadores (x86) (Lopes, 2011).

A empresa MikroTik desempenha um papel crucial em prover soluções eficientes para provedores de serviços de Internet, empresas e organizações em todo o mundo devido à sua oferta de equipamentos avançados e acessíveis. São mais de 500 distribuidores oficiais e revendedores em 145 países (Dicomp, 2020).

[3] https://mikrotik.com/software
[4] https://mikrotik.com/
[5] https://mikrotik.com/products

3 METODOLOGIA

A seguir é descrita a metodologia e tecnologias utilizadas para a execução das atividades relacionadas aos objetivos específicos citados anteriormente destacando as contribuições práticas do software desenvolvido.

3.1 DEFINIÇÃO DE PESQUISA

Este trabalho, sob o ponto de vista de sua natureza, é caracterizado como uma pesquisa aplicada uma vez que objetiva gerar conhecimentos para aplicação prática, direcionados à solução de problemas específicos.

> "[...] procedimento racional e sistemático que tem como objetivo proporcionar respostas aos problemas que são propostos. A pesquisa desenvolve-se por um processo constituído de várias fases, desde a formulação do problema até a apresentação e discussão dos resultados" (Gil, 2008, p. 17)

O planejamento sequencial dos procedimentos empregados no desenvolvimento da pesquisa abarca a etapa inicial da investigação científica. Essa fase abrange desde a seleção do tópico de estudo até a concretização prática dos métodos utilizados

3.2 TIPO DE PESQUISA

A pesquisa realizada neste trabalho se enquadra na categoria de pesquisa exploratória, cujo propósito é fornecer critérios, compreensão e *insights* sobre conceitos e aspectos pouco conhecidos. Estudos exploratórios desempenham frequentemente um papel crucial na identificação de cenários, na busca por alternativas e no estímulo à descoberta de novas ideias (Zikmund, 2000).

De acordo com Selltiz *et al.* (1965), em pesquisas exploratórias, nem sempre é necessário formular hipóteses. Isso permite que o pesquisador amplie seu

conhecimento sobre os fatos, o que facilita a formulação precisa de problemas, o surgimento de novas hipóteses e a realização de pesquisas mais estruturadas. Nessas circunstâncias, o planejamento da pesquisa deve ser suficientemente flexível para analisar diversos aspectos relacionados ao fenômeno.

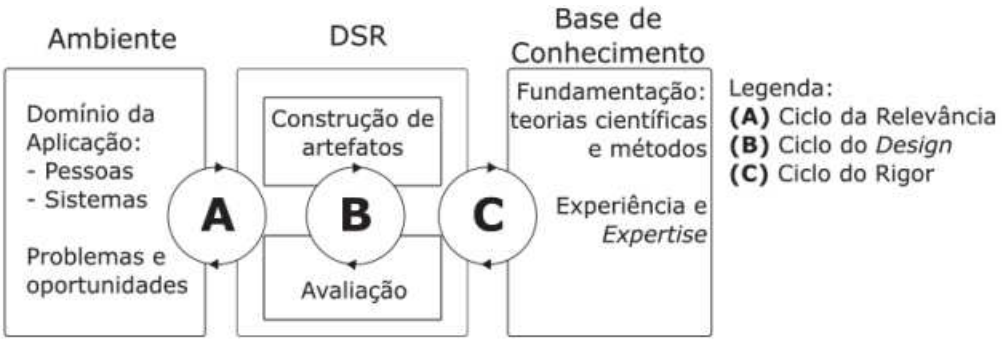
3.3 DEFINIÇÃO DE PESQUISA

Este trabalho seguiu os passos definidos pela metodologia DSR (*Design Science Research*). Esta metodologia é amplamente utilizada em disciplinas da área de Ciência da Computação, Sistemas de Informação e Engenharia de Software. Seu objetivo principal é projetar e desenvolver soluções para problemas práticos, frequentemente relacionados a sistemas de informação e inovação. Ela enfatiza que os pesquisadores não apenas estudam problemas, mas também contribuem para sua resolução, projetando soluções eficazes. A abordagem DSR segue um ciclo iterativo, que compreende as seguintes etapas:

- Identificação do Problema e motivação;
- Definição dos Requisitos;
- Design da Solução;
- Desenvolvimento e Implementação;
- Avaliação e Validação;
- Comunicação dos Resultados;
- Refinamento Iterativo (se necessário).

No que tange a relação entre a prática e a teoria, Hevner e Chatterjee (2010) propõem uma representação do método DSR em três ciclos: etapa de relevância, na qual o pesquisador procura informações sobre o problema em questão; o ciclo de design, voltado para o desenvolvimento e avaliação do artefato; e o ciclo de rigor, no qual se busca a fundamentação teórica, enfatizando as contribuições substanciais do estudo (*apud* Davila; Reis, 2010, p. 3). A Figura 12 ilustra os ciclos citados.

Figura 12 - Ciclos do DSR



Fonte: Hevner; Chatterjee (2010)

O Quadro 2 apresenta as etapas da metodologia preenchidas:

Quadro 2 - Metodologia DSR

Identificação do problema	Realizar uma análise das necessidades dos administradores de TI e equipes de segurança no gerenciamento de redes.
Definição dos Requisitos	Coletar métricas de hardware via API RouterOS, gerenciar configurações de equipamentos Mikrotik, analisar cenários para detecção de eventos.
Design da Solução	Coleta de dados de equipamentos com sistema operacional RouterOS via API utilizando a linguagem de programação Python e banco de dados MongoDB.
Desenvolvimento e Implementação	Desenvolver métodos eficientes para coletar e analisar métricas de uso de hardware e status de conexões via API RouterOS.
Avaliação e Validação	Estabelecer critérios para avaliar a eficácia do DataRouter, como precisão na detecção de eventos, facilidade de uso e capacidade de detectar incidentes.
Comunicação dos Resultados	Destacar as contribuições do estudo para o gerenciamento eficaz da infraestrutura de rede, prevenindo falhas e mitigando prejuízos financeiros, perda de clientes e comprometimento de informações do banco de dados.
Refinamento Iterativo	Reutilização do processo de pesquisa, se necessário, utilizando os resultados da análise das predições para refinar e aprimorar as formas de visualização

Fonte: Elaborado pelo autor

3.4 CENÁRIO ANALISADO

O cenário examinado pelo software DataRouter constitui um estudo de caso centrado no equipamento MikroTik modelo RB3011UiAS-RM, implantado na sede de uma empresa localizada no Extremo-Sul de Santa Catarina. Para realizar a coleta de dados, foram estabelecidos protocolos e critérios específicos de interação com a API garantindo a captura de informações relevantes e um monitoramento eficaz.

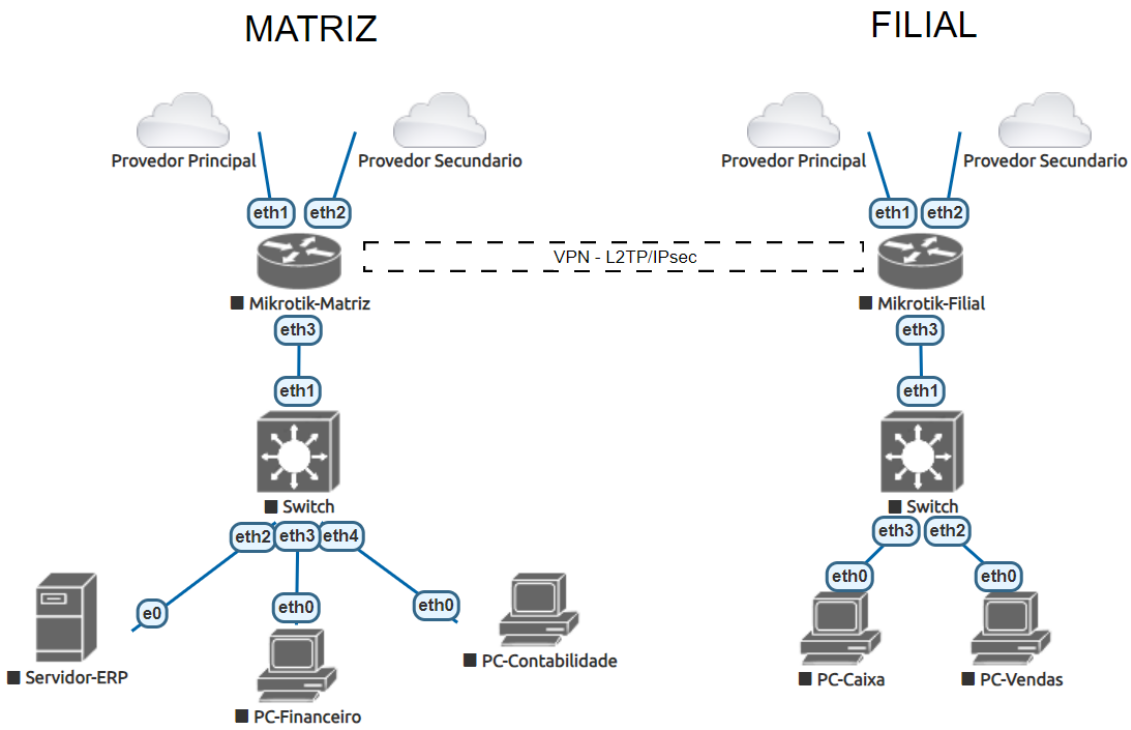
O sistema foi implantado utilizando um servidor Windows Server 2022 Standard, com licenciamento obtido por meio da parceria entre a UFSC e o Microsoft Dev Tools[6].

A topologia da rede utilizada para este estudo, conta com dois *links* de Internet e conexões de VPN com suas filiais. Desta forma, é possível nesse cenário e com os recursos disponíveis no software desenvolvido, acompanhar quedas de conexão com os provedores de internet, quedas de conexão de VPN com a filial, tráfego na rede além de dados internos do dispositivo. As principais funções do equipamento analisado são:

- Gerenciamento de Rotas;
- Gerenciamento de sub-redes com VLANs *(***Virtual Local** Area Network);
- *Failover* entre provedores de internet;
- Firewall;
- Servidor DHCP;
- Servidor DNS;
- Servidor L2TP.

[6] https://azureforeducation.microsoft.com/devtools

Figura 13 - Topologia de Rede



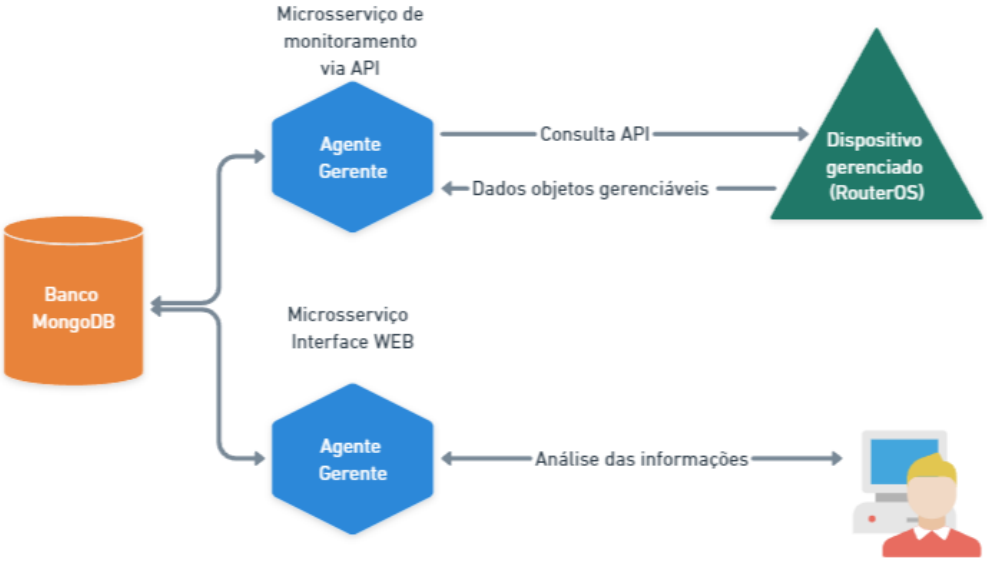
Fonte: Elaborado pelo autor

3.5 COLETA E PREPARAÇÃO DOS DADOS

A fim de extrair as informações essenciais do dispositivo monitorado, é de fundamental importância analisar quais requisitos computacionais dos equipamentos monitorados. Essa definição possibilita tomar decisões direcionadas à evolução do software, garantindo seu desempenho e funcionalidade. Sendo assim, é possível otimizar a gestão e garantir que as necessidades do monitoramento sejam atendidas de forma eficiente.

Sendo assim, como o sistema operacional RouterOS pode ser utilizado em equipamentos com até 32MB de memória RAM e 64MB armazenamento interno, foi utilizado o tipo de topologia de gerência centralizada devido a sua simplicidade, eficiência operacional e centralização das responsabilidades em um único dispositivo com maior poder computacional.

Figura 14 - Topologia de gerência do Projeto DataRouter



Fonte: Elaborado pelo autor

3.5.1 Objetos Gerenciáveis

Através da análise dos dados coletados é possível identificar padrões, tendências e anomalias, permitindo tomar decisões embasadas em dados sólidos. Essa abordagem contribui para uma tomada de decisão ou decisões mais assertivas e estratégicas, resultando em melhorias contínuas no sistema e na rede monitorada.

Dessa forma, diante da ampla gama de informações acessíveis na API do sistema RouterOS durante o processo de desenvolvimento do software, foram coletados os objetos gerenciáveis que exercem maior impacto na rede em análise. Essa abordagem permitiu uma análise mais precisa e focada nos elementos cruciais para o funcionamento e desempenho da rede. Ao priorizar os objetos com maior influência, foi possível direcionar os esforços para otimizar a gestão e aprimorar a eficiência do sistema como um todo. Dentro dos dados coletados estão:

- Tráfego de rede de cada interface física ou virtual;
- *Status* de conexão de cada interface física ou virtual;
- Número de quedas de conexão em cada interface física ou virtual;
- Data e Hora da última queda de conexão de cada interface física;
- Uso de CPU;
- Frequência da CPU;
- Temperatura CPU;
- Uso da memória RAM;
- Uso de armazenamento da memória interna;
- *Neighbor Dicovery* (MNDP) e LLDP;
- Latência de servidores DNS.

3.5.2 Alertas

O sistema DataRouter permite criar vários alertas e gatilhos no seu código fonte, para este trabalho foram desenvolvidos os seguintes alertas para o usuário:

- Uso de acima de CPU 75%;
- Uso de acima em 100%;
- temperatura de CPU;
- Teste de Ping com perda de pacotes acima de 25%;
- Teste de Ping com perda de pacotes acima de 50%;
- Teste de Ping com perda de pacotes acima de 75%;
- Atualização disponível;

4 PROJETO DE SOFTWARE

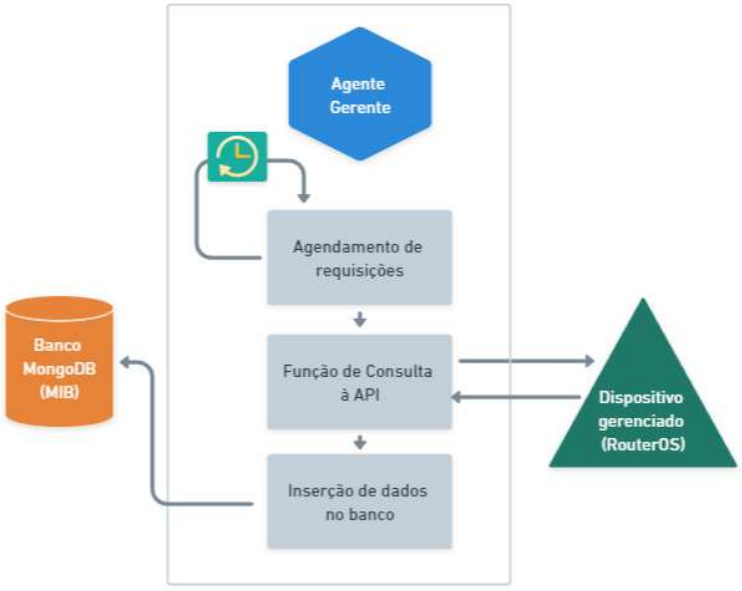
Esta seção apresentará de forma mais detalhada as características do *software* desenvolvido. Serão abordados o levantamento de pré-requisitos do equipamento com sistema RouterOS, definição de estrutura de monitoramento, forma de coleta de dados e outros aspectos relevantes para o processo.

A aplicação foi desenvolvida em dois microsserviços distintos. Pois, esta estrutura possibilita que a aplicação seja desenvolvida, testada e implantada de forma independente. Além disso, ao segmentar as responsabilidades de monitoramento é criado uma camada de segurança.

O primeiro microsserviço, desenvolvido em Python, desempenha a função de coletar dados de objetos gerenciados por meio de uma API. Ele também é encarregado de agendar os intervalos nos quais as requisições devem ocorrer, destacando que objetos mais críticos necessitam de uma frequência de verificação mais elevada. Por fim, os dados são armazenados em um banco de dados não relacional MongoDB.

Conforme demonstrado na figura 15, a fim de evitar que o próprio monitoramento gere uma sobrecarga no equipamento. É seguindo um limite de requisições por segundo de acordo com a relevância dos dados conforme quadro 2.

Figura 15 - Estrutura microsserviço responsável pela consulta à API



Fonte: Elaborado pelo autor

Quadro 3 - Tempo entre requisições

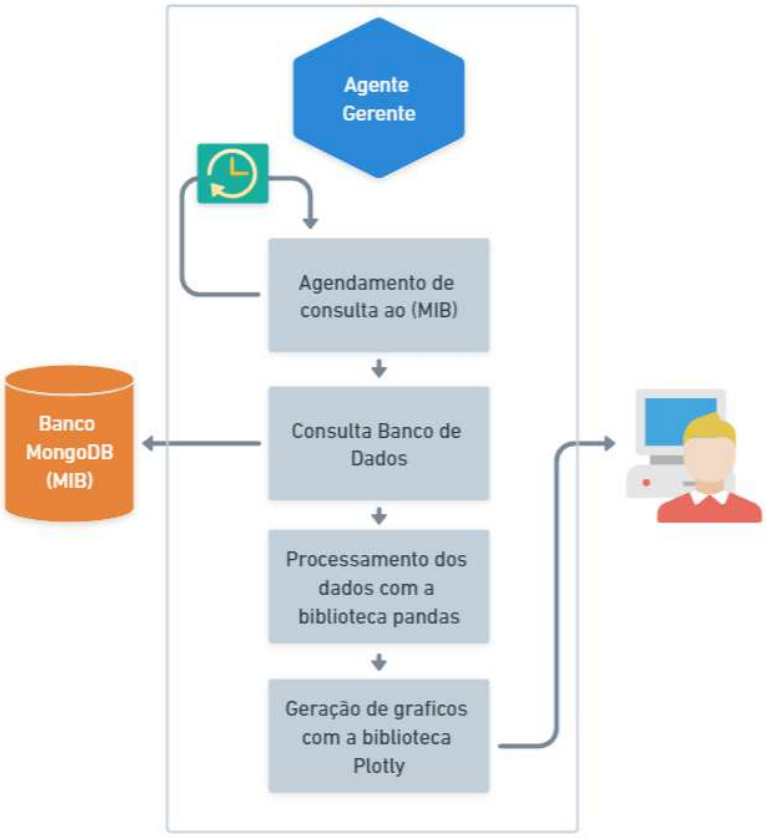
Função executada	**Intervalo**
Status do sistema	2 segundos
Monitoramento de tráfego	1 segundos
Atualização de novas Interfaces	120 segundos
Atualização dispositivos vizinhos	30 segundos
Teste de Ping com DNS	60 segundos
Atualização de rotas	120 segundos

Fonte: Elaborado pelo autor

Consequentemente, o segundo microsserviço é responsável pelo processamento dos dados armazenados no MIB e disponibilização dos dados em uma interface WEB apresentado para o usuário em gráficos e planilhas. Através dessa interface, o usuário pode explorar os dados de forma interativa e obter percepções valiosas na tomada de decisões gerenciais.

Conforme demonstrado na figura 16 do microsserviço o agente gerente faz novas consultas no banco e atualiza os gráficos a cada 5 segundos.

Figura 16 - Estrutura microsserviço responsável pela análise dos dados



Fonte: Elaborado pelo autor

4.1 BIBLIOTECAS UTILIZADAS

Entre as bibliotecas mais importantes para o desenvolvimento do *software* destacam-se o PyMongo, o Pandas, o Dash, o *Dash Bootstrap Components* e o Plotly conforme quadro 3.

Quadro 4 - Principais bibliotecas Python utilizadas

Biblioteca Python	**Função**
pymongo.py	Conexão no banco de dados MongoDB
pandas.py	Manipulação de dados
dash.py	Interface WEB
plotly.py	Visualização de gráficos e planilhas

Fonte: Elaborado pelo autor

O PyMongo é uma biblioteca que permite conectar e interagir com bancos de dados MongoDB, tornando o gerenciamento de dados não estruturados uma tarefa simples. O Pandas é uma poderosa biblioteca para manipulação e análise de dados tabulares, oferecendo ferramentas para carregar, limpar e processar dados de forma eficaz.

O Dash, em conjunto com o *Dash Bootstrap Components* e o Plotly, possibilita a criação de aplicativos web interativos e painéis de visualização de dados. O Dash permite criar aplicativos web com facilidade, enquanto o *Dash Bootstrap Components* oferece componentes de interface de usuário elegantes e o Plotly facilita a criação de gráficos e visualizações impressionantes.

4.2 ESTRUTURA DE DADOS DE UM SISTEMA DE GERENCIAMENTO

A estrutura do MIB (*Management Information Base*) foi desenvolvida no banco de dados não relacional MongoDB, devido à flexibilidade de iniciar o projeto, uma vez que esta arquitetura não exige a conformidade com um formato pré-definido. Sendo assim, proporcionando a capacidade de lidar com informações de maneira dinâmica.

No banco de dados MongoDB cada unidade de informação é armazenada em um documento com a notação JSON. Portanto, como a requisição a API do RouterOS

retorna os dados no formato JSON é possível inserir os dados diretamente no banco com muita praticidade. A figura 17 mostra um trecho de código da função que insere no banco de dados as informações sobre as interfaces do dispositivo.

Figura 17 - Código demonstrado a inserção da requisição à API no banco

```
def interfaces():

    collection = database['rb_interface']

    try:
        response = router.talk('/interface/print')
        response = [response]
        data_dict = response[0]

        # Converter a lista em um data frame
        df = pd.DataFrame(data_dict)
        df["data"] = datetime.now().strftime("%Y-%m-%d")
        df["hora"] = datetime.now().strftime("%H:%M")
        df["data-hora"] = datetime.now().isoformat()

        # Converter o data frame em um formato compatível com JSON
        df_json = df.to_dict(orient='records')
        collection.insert_many(df_json)
        print("Teste interface finalizado")

    except router.LoginError(Exception) as e:
        print(f"Ocorreu um erro na requisição: {e}")
```

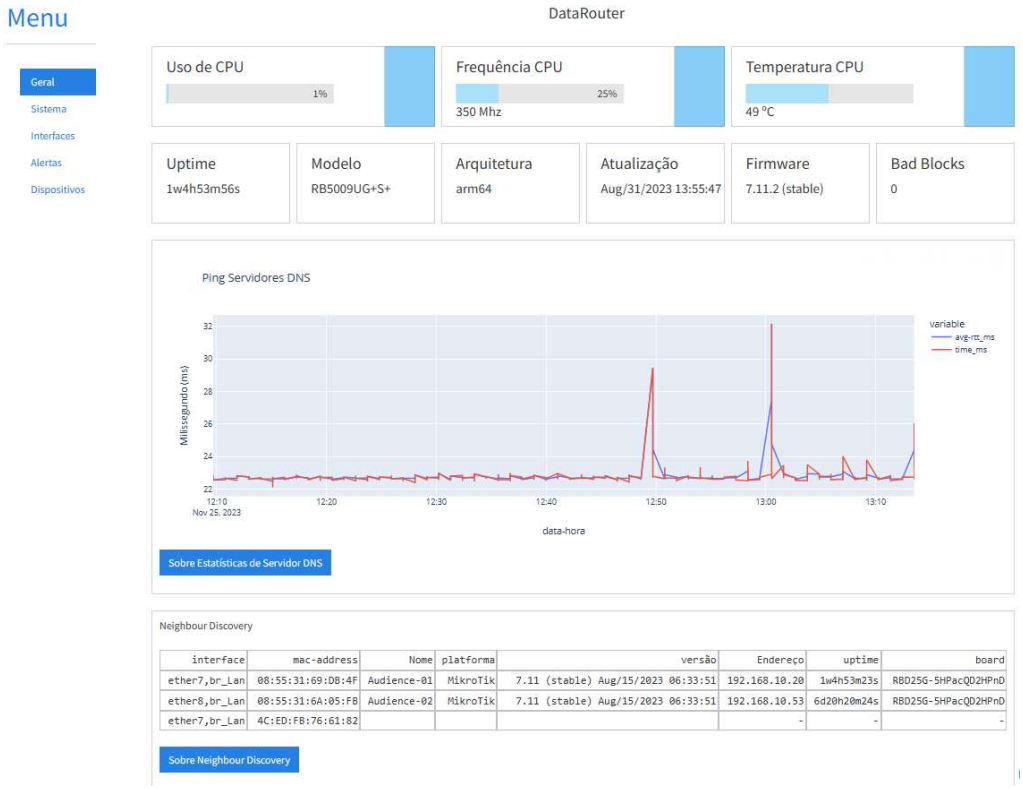
Fonte: Elaborado pelo autor

4.3 INTERFACES DE MONITORAMENTO

O software DataRouter conta com cinco menus na sua barra lateral esquerda, esses menus abrigam quatro telas com informações referentes ao monitoramento e uma para a conexão com o dispositivo que está sendo monitorado, sendo as seguintes opções:

- Geral - Principais informações da rede;
- Sistema - Principais informações do dispositivo monitorado;
- Interfaces - Informações das interfaces do dispositivo monitorado;
- Alertas - Histórico de Alertas;
- Dispositivos - Configuração de acesso a API do dispositivo monitorado.

Figura 18 – Tela principal do sistema DataRouter

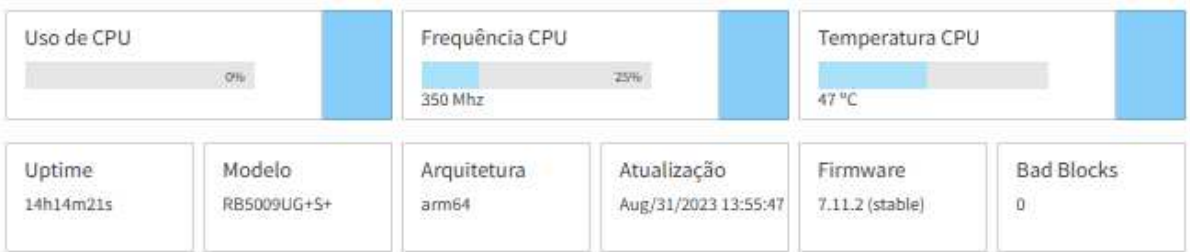


Fonte: Elaborado pelo autor

4.3.1 INDICADORES DO MENU GERAL

O primeiro elemento gráfico contém informações sobre os recursos da CPU e hardware e tem a capacidade de alertar o usuário sobre falhas de hardware, limitações inerentes ao dispositivo, erros de configuração e possíveis ataques. Adicionalmente, apresenta dados básicos, como o tempo de atividade do dispositivo, versões de software e de hardware.

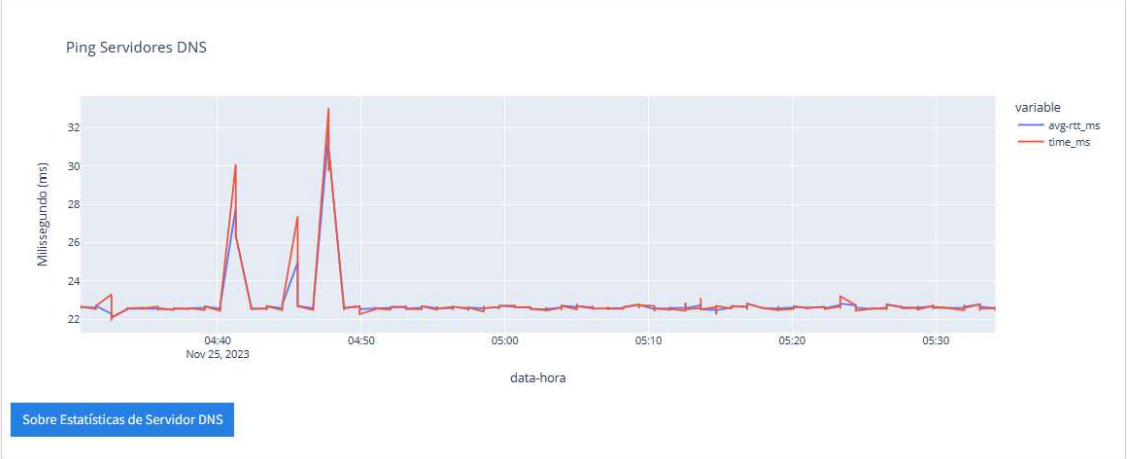
Figura 19 - *Dashboard* da tela principal do Software



Fonte: Elaborado pelo autor

O gráfico de latência de *ping* associado ao servidor DNS desempenha duas funções essenciais. Primeiramente, é utilizado para monitorar a qualidade da rede e a latência, sendo capaz de identificar possíveis problemas relacionados à perda de pacotes ou rotas. Em segundo lugar, é útil para detectar quedas de conexão com o provedor de internet. Vale ressaltar que, em comparação, é mais provável que ocorram interrupções no *link* fornecido pelo provedor de internet do que no servidor DNS do Google.

Figura 20 - Gráfico de latência de *ping* com servidor de DNS



Fonte: Elaborado pelo autor

A tabela de *Neighbour Discovery* lista os dispositivos encontrados na rede compatíveis com o protocolo *MikroTik Neighbour Discovery* (MNDP), *Link Layer Discovery Protocol* LLDP ou CDP (*Cisco Discovery Protocol*) ou LLDP no domínio de transmissão Layer2.

Figura 21 - Tabela com lista de dispositivos vizinhos encontrados

Neighbour Discovery

interface	mac-address	Nome	platforma	versão	Endereço	uptime	board
ether7,br_Lan	08:55:31:69:DB:4F	Audience-01	MikroTik	7.11 (stable) Aug/15/2023 06:33:51	192.168.10.20	1w21h38m24s	RBD25G-5HPacQD2HPnD
ether8,br_Lan	08:55:31:6A:05:FB	Audience-02	MikroTik	7.11 (stable) Aug/15/2023 06:33:51	192.168.10.53	1w21h37m22s	RBD25G-5HPacQD2HPnD
ether8,br_Lan	08:55:31:6A:05:FB	Audience-02	MikroTik	7.11 (stable) Aug/15/2023 06:33:51	192.168.10.53	1w21h37m22s	RBD25G-5HPacQD2HPnD

Sobre Neighbour Discovery

Fonte: Elaborado pelo autor

O gráfico de tráfego da rede pode ajudar a identificar gargalos e problemas de desempenho. Pois, ao analisar os picos e vales no tráfego, é possível determinar se há congestionamento em determinadas interfaces, ajudando na resolução proativa de problemas. Pois, picos de tráfego podem indicar atividades maliciosas, como ataques ou tentativas de intrusão.

Além disso, ao analisar os dados de tráfego ao longo do tempo, é possível realizar um planejamento mais eficaz para a expansão da rede. Isso inclui a adição de largura de banda ou a implementação de políticas de gerenciamento de tráfego para acomodar o crescimento futuro.

Figura 22 - Gráfico tráfego da rede por interface

Fonte: Elaborado pelo autor

4.3.2 INDICADORES DO MENU SISTEMA

A segunda tela tem uma visão detalhada das informações relacionadas ao uso e frequência do CPU e memória RAM disponível e armazenamento interno, esta tela tem como objetivo detectar picos do uso de recursos do dispositivo. Além disso, permite antecipar eventos como a aproximação do limite de capacidade do equipamento monitorado.

Figura 23 - Gráfico de monitoramento do uso de Hardware

Fonte: Elaborado pelo autor

Em suma, essa tela fornece uma visão técnica e detalhada do estado operacional do dispositivo, capacitando os usuários a adotarem medidas proativas com base em informações precisas sobre o desempenho do hardware e a utilização de recursos críticos.

4.3.3 INDICADORES DO MENU INTERFACES

A terceira tela informa o *status* de conexão das interfaces físicas e virtuais do equipamento, sendo possível também acompanhar a quantidade de quedas de conexão e data da última queda.

Ao visualizar o status das interfaces físicas, os usuários podem identificar imediatamente se há alguma desconexão ou problema na conectividade de hardware. A análise das interfaces virtuais, por sua vez, fornece insights sobre a integridade das conexões.

Figura 24 - Status Interfaces monitoradas

Nome	Tipo	Quedas de Link	Última queda	data-hora	Desabilitada	Status
ether1	ether	0	-	2023-07-03T01:27:54.782463	não	OK
ether2	ether	0	-	2023-07-03T01:27:54.782463	não	Sem Conexão
ether3	ether	0	-	2023-07-03T01:27:54.782463	não	Sem Conexão
ether4	ether	0	-	2023-07-03T01:27:54.782463	não	Sem Conexão
ether5	ether	0	-	2023-07-03T01:27:54.782463	não	OK
ether6	ether	5	jun/03/2023 12:55:07	2023-07-03T01:27:54.782463	não	OK
ether7	ether	2	may/25/2023 01:22:55	2023-07-03T01:27:54.782463	não	OK
ether8	ether	2	may/25/2023 01:22:56	2023-07-03T01:27:54.782463	não	OK
sfp-sfpplus1	ether	0	-	2023-07-03T01:27:54.782463	sim	Sem Conexão
-	bridge	0	-	2023-07-03T01:27:54.782463	não	OK
-	bridge	0	-	2023-07-03T01:27:54.782463	não	OK
-	pppoe-out	6	jul/02/2023 03:03:07	2023-07-03T01:27:54.782463	não	OK
-	zerotier	0	-	2023-07-03T01:27:54.782463	sim	Sem Conexão
-	veth	0	-	2023-07-03T01:27:54.782463	não	OK

Fonte: Elaborado pelo autor

4.3.4 INDICADORES DO MENU ALERTAS

A quarta tela contém uma listagem com dados dos últimos eventos registrados, sendo possível visualizar alertas em que o operador não visualizou em tempo real.

Figura 25 - Eventos e alertas registrados

Últimos alertas

Alertas	Criticidade	Data	Hora
USO CPU 100%	Alta	20/11/2023	02:35:10
USO CPU 75%	Média	21/11/2023	10:09:59
Atualização Disponível	Baixa	23/11/2023	14:20:34
Temperatura CPU elevada	Alta	01/12/2023	01:41:10
Teste de ping (50 % pacotes perdidos)	Média	01/12/2023	12:20:33
Teste de ping (50 % pacotes perdidos)	Média	01/12/2023	12:22:41

Fonte: Elaborado pelo autor

Ao revisar essa lista, os operadores têm a capacidade de retroceder no tempo e entender o contexto de eventos passados, identificando padrões ou correlações que podem ter impacto nas operações do dispositivo monitorado.

Além disso, a capacidade de visualizar eventos não observados em tempo real oferece uma camada adicional de segurança e garantia de que nenhum alerta crítico seja negligenciado. Essa abordagem proativa permite que os operadores estejam

cientes de eventos passados relevantes, mesmo que não tenham sido detectados no momento da ocorrência.

4.3.5 MENU DISPOSITIVOS

Na tela de configuração da conexão com a API MikroTik, conta com cinco campos de inserção — descrição, IP do dispositivo, porta de acesso da API, usuário e senha — os administradores podem configurar de maneira direta o dispositivo MikroTik a ser monitorado.

Figura 26 - Cadastro de acesso ao dispositivo monitorado

Fonte: Elaborado pelo autor

5 RESULTADOS E DISCUSSÃO

Este capítulo apresenta os resultados e discussões obtidos através do desenvolvimento da ferramenta DataRouter. A ferramenta desenvolvida mostrou-se eficaz no monitoramento contínuo e a coleta de informações dos equipamentos.

A utilização da linguagem Python e a integração com a API mostraram-se adequadas para o desenvolvimento dessa ferramenta, permitindo a comunicação eficiente com os roteadores com sistema operacional RouterOS. Além disso, a escolha de uma arquitetura centralizado demonstrou-se eficaz no cenário analisado e durante o tempo de análise foi capaz de detectar de forma reativa uma queda do link de internet que era utilizado como contingência. Esse alerta é essencial, pois quando o Link de internet principal tem algum problema é esperado que o secundário esteja operando. E foi possível perceber melhorias no tempo de atendimento e na resolução de problemas.

Os resultados obtidos evidenciaram a importância do monitoramento contínuo da rede e a utilização de ferramentas adequadas para esse fim. O software desenvolvido demonstrou ser uma solução eficiente e prática para auxiliar os profissionais de TI na gestão e manutenção da rede de computadores.

6 CONCLUSÃO

O objetivo desta pesquisa, foi analisar a importância do gerenciamento de redes e com foco em roteadores que utilizam o sistema operacional RouterOS desenvolvimento de soluções com os conhecimentos do curso de TIC. Para isso, observando-se o ambiente de uma rede corporativa.

Diante de tal afirmação, o presente trabalho demonstrou a importância do conhecimento dos tipos de gerenciamento de redes e as cinco áreas do modelo de gerência de redes OSI, já que com base nesse conhecimento foi possível de forma assertiva escolher as melhores soluções para gerenciamento para esse projeto.

Depois, foi abordado o desenvolvimento do software de monitoramento, onde a escolha da gerência de redes centralizada junto com a linguagem Python e banco de dados MongoDB mostraram-se uma boa escolha nesse projeto diante da facilidade em criar os documentos que armazenam dados, sem a necessidade de ter uma arquitetura do banco pré-definida.

Com base na comparação do histórico dos dados, foi possível detectar anomalias da linha de base. Como por exemplo tráfego inesperado, uso de CPU, memória RAM e *logs* de dispositivos tentando acessar o equipamento por fim, esta ferramenta desenvolvida se demostra muito importante, pois além de indicar falhas no equipamento ou anomalias na rede que permita identificar de maneira precoce de um ataque cibernético ou *ransomware*.

Espera-se que os resultados deste estudo contribuam para a compreensão dos benefícios do monitoramento da infraestrutura de rede e incentivem a adoção dos conceitos e estratégias para melhorar a segurança na infraestrutura da rede de computadores e proteger os usuários de falhas que possam impactar seus negócios.

Quanto às limitações da pesquisa, reforça-se a necessidade que em um cenário para um uso profissional é necessário capturar mais informações, adicionar mais elementos visuais e configurações, a fim de tornar a ferramenta mais flexível e completa para esse uso.

Sugere-se então, com base nessa pesquisa, para trabalhos futuros utilizar a base de dados fornecida pelo monitoramento e com o treinamento de uma inteligência artificial para criar avisos de forma preditiva, além de analisar a possibilidade de utilizar

o banco de dados *elasticsearch* por ser mais indicado como mecanismos de análise de dados.

Diante disso, o presente trabalho teve o objetivo alcançado em relatar a importância do monitoramento de redes de computadores com estudo de caso em roteadores com sistema operacional RouterOS. Possivelmente, estudo ainda poderá auxiliar administradores de redes que necessitem de um estudo detalhado sobre as ferramentas analisadas.

ANEXOS

Anexo A

Link para código fonte microsserviço responsável pela consulta à API:
<https://github.com/juliocesarbiz/Monitor-API-Mikrotik>

Anexo B

Link para código fonte microsserviço responsável pela análise dos dados:
<https://github.com/juliocesarbiz/DashRouterOS>

REFERÊNCIAS

BATTISTI, Gerson. **Modelo de Gerenciamento para Infraestruturas de Medições de desempenho em redes de computadores.** Gerson Battisti – Porto Alegre: Programa de Pós-Graduação em Computação, 2007. 131p. Universidade Federal do Rio Grande do Sul. Disponível em: <https://www.lume.ufrgs.br/bitstream/handle/10183/12671/000632788.pdf;sequence=1>. Acesso em: 14 abr 2023.

BIANCHINI, A. **Gerenciamento de Rede**. 29 dez 2016. Apresentação do Power Point. Disponível em: <https://silo.tips/download/gerenciamento-de-rede-alessandro-c-bianchini>. Acesso em: 16 abr 2023.

BISOL, Gabriel Alves. **DESENVOLVIMENTO DE UMA SOLUÇÃO PARA TROCA DE DADOS ENTRE UM ERP E UM APLICATIVO MÓVEL.** Gabriel Alves Bisol. Caxias do Sul, 2019. 82p. TCC. Universidade de Caxias do Sul. Disponível em: <https://repositorio.ucs.br/xmlui/bitstream/handle/11338/5965/TCC%20Gabriel%20Alves%20Bisol.pdf?sequence=1&isAllowed=y>. Acesso em: 30 abr 2023.

BORGES, Luiz Eduardo. **Python para Desenvolvedores**. São Paulo: Novatec Editora, 2014.

CASTELLS, Manuel. **A galáxia da internet: reflexões sobre a internet, os negócios e a sociedade**. Rio de Janeiro: Editora Paz e Terra, 2003.

CICHACZEWSKI, João Carlos. **ANÁLISE DA INTERFERÊNCIA MÚTUA ENTRE REDES IEEE 802.11 E IEEE 802.15.4**. João Carlos Cichaczewski. Araranguá, 2013. 107p. TCC. Universidade Federal de Santa Catarina.

DAVILA, Nicole C.; REIS, Adriana N. dos. **Construção de aplicações computacionais na saúde: explorando a abordagem Design Science Research**. *In*: ESCOLA REGIONAL DE COMPUTAÇÃO APLICADA À SAÚDE (ERCAS), 6, 2018, Niterói. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2018.

DANTAS, L. M. **Segurança da informação**: **uma abordagem focada em gestão de riscos**. Olinda: Livro Rápido, 2011.

DICOMP, **Routerboard um mundo de possibilidades para gerenciamento de redes,** nov. 2023. Disponível em: **<**https://www.dicomp.com.br/noticia-/155/routerboard-um-mundo-de-possibilidades-para-gerenciamento-de-redes/>. Acesso em: 01 dez 2023.

ELER, Esdras de O. **Modelo TMN: Aplicação ao Gerenciamento de Redes de Telecomunicações**, out. 2015. Disponível em: <https://www.teleco.com.br-/tutoriais/tutorialmodelotmn/>. Acesso em: 14 maio 2023.

FERNANDES, Carlos Rafael Magalhães. **Integração de Dados para Compartilhamento de Informações Relacionadas aos Serviços Disponíveis em Unidades de Saúde.** Carlos Rafael Magalhães Fernandes. Criciúma, 2019. 74p. TCC. Universidade do Extremo Sul Catarinense. Disponível em: <http://repositorio.unesc.net/bitstream/1/8200/1/CARLOS%20RAFAEL%20MAGALH%C3%83ES%20FERNANDES.pdf>. Acesso em: 30 abr 2023.

FOROUZAN, Behrouz A. **Comunicação de Dados e Redes de Computadores**, 3.ed, Tradução de Glayson Eduardo de Figueiredo, Porto Alegre, Bookman, 2006.

GIL, Antonio Carlos. **Como elaborar projetos de pesquisa**. 4. ed. São Paulo: Atlas, 2008.

Hevner, A.; Chatterjee, S. **Design Research in Information Systems: Theory and Practice**. Flórida: Springer US, 2010.

KARL Kurbel, T. S**. Integration issues of information engineering based i-case tools**. Working Papers of the Institute of Business Informatics. 1994.

KUROSE, JAMES F.; ROSS, KEITH W. **Redes de Computadores e a Internet: uma abordagem top-down** São Paulo: Editora Pearson Addison Wesley, 3ª Edição, 2006. 632 p.

LYRA, Maurício R. **Segurança e auditoria em sistemas de informação**. Rio de Janeiro: Ciência Moderna, 2008.

MACEDO, Ricardo Tombesi. **Redes de computadores**. Ricardo Tombesi Macedo ... [et al.]. – 1. ed. – Santa Maria, Universidade Federal de Santa Maria, NTE, 2018. 196p. Acesso em:<https://repositorio.ufsm.br/bitstream/handle/1/18351/Curso_Lic-Comp_Redes-Computadores.pdf?sequence=1&isAllowed=y>. Acesso em: 16 abr 2023.

MCKINNEY, Wes. **Python para Análise de Dados: Tratamento de dados com Pandas, NumPy e IPython**. São Paulo: Novatec Editora, 2011.

MORAES, **Alexandre Fernandes de. Redes de Computadores Fundamentos**. São Paulo: Saraiva Educação S.A., 2020. ISBN: 9788536509839 (livro digital).

NETO, José André Carneiro. **Um mapeamento de práticas em projetos de APIs REST**. José André Carneiro Neto. Recife, 2020. 62p. TCC. Universidade Federal de Pernambuco Centro de Informática. Disponível em: <https://www.cin.ufpe.-br/~tg/2020-3/TG_CC/tg_jacn.pdf>. Acesso em: 30 abr 2023.

OLIVEIRA, Álvaro Gabriel Gomes de. **Construção de aplicações distribuídas utilizando-se de APIs REST**. Álvaro Gabriel Gomes de Oliveira - Mossoró, 2018. 64p. Monografia (Graduação). Universidade do Estado do Rio Grande do Norte. Disponível em: <https://di.uern.br/tccs2019/html/ltr/PDF/014006456.pdf>. Acesso em: 30 abr 2023.

ORACLE. **What Is a Socket?** Disponível em: <https://docs.oracle.com/javase/tutorial/networking/sockets/definition.html>. Acesso em: 06 nov 2023.

PERIN, Christian. **O que são e como funcionam os sockets. 2022**. Disponível em: https://mydatabase.com.br/index.php/home/75-categorias/sistemas-operacionais/protocolos/socket/228-o-que-s%C3%A3o-e-como-funcionam-os-sockets. Acesso em: 13 nov. 2023.

ROSA, Thiago Pereira **Um método para o desenvolvimento de software baseado em microsserviços**. Thiago Pereira Rosa. – Quixadá, 2016. 64p. Monografia (graduação) – Universidade Federal do Ceará. Disponível em: <https://repositorio.-ufc.br/bitstream/riufc/25123/1/2016_tcc_tprosa.pdf>. Acesso em: 16 abr 2023.

SELLTIZ, C.; WRIGHTSMAN, L. S.; COOK, S. W. **Métodos de pesquisa das relações sociais**. São Paulo: Herder, 1965.

SILVA, Firmino Oliveira da. **Integração de sistemas e plataformas como solução para a gestão da informação de clientes**. 2004. 215p. Dissertação (Mestrado). Universidade do Porto. Disponível em: <https://repositorio-aberto.up.pt/bitstream/10216/11378/2/Texto%20integral.pdf>. Acesso em: 10 maio 2023.

SPECIALSKI, Elizabeth S. **Apostila de Gerência de Redes de Computadores e Telecomunicações**. CPGCC: Florianópolis, 2000.

SOUSA, Lindenberg Barros de. **Gerenciamento e segurança de redes**. São Paulo: SENAI-SP Editora, 2017.

TEDESCO, Kennedy. **Uma introdução a TCP, UDP e Sockets**. 2019. Disponível em: <https://www.treinaweb.com.br/blog/uma-introducao-a-tcp-udp-e-sockets >. Acesso em: 06 nov. 2023.

UFSC - Universidade Federal de Santa Catarina. Cursos de Graduação: **Bacharelado em Tecnologias da Informação e Comunicação**. Disponível em: <https://-ararangua.ufsc.br/cursos-de-graduacaobacharelado-em-tecnologias-da-informacao-e-comunicacao/>. Acesso em: 25 nov. 2023.

LÓSCIO, F.; OLIVEIRA, D.; PONTES, S. NoSQL no d**esenvolvimento de aplicações web colaborativas. In: VIII Simpósio Brasileiro de Sistemas Colaborativos**, Brasil, 2011. Disponível em: <www.addlabs.uff.br/sbsc_site/SBSC2011_NoSQL.pdf>. Acesso em: 15 jun. 2023.

LOPES, Rafael Carlos. **Você conhece o RouterOS Mikrotik?** 2011. Disponível em: <https://www.vivaolinux.com.br/artigo/Voce-conhece-o-RouterOS-Mikrotik?pagina=1>. Acesso em 01 dez. 2023.

MAZIERO, Carlos. Comunicação em Rede. 2008. Disponível em: **<https://wiki.inf.ufpr.br/maziero/doku.php?id=pua:comunicacao_em_rede**>. Acesso em: 01 nov. 2023.

MONGODB. **O que é o MongoDB?** [S.d.]. Disponível em: <https://www.mongodb-.com/pt-br/what-is-mongodb>. Acesso em: 10 nov. 2023.

MikroTik. **RouterOS Documentation**. Disponível em: **<https://help.mikrotik.com-/docs/display/ROS/API>**. Acesso em: 10 ago. 2022.

ZIKMUND, W. G. **Business research methods.** 5.ed. Fort Worth, TX: Dryden, 2000.