UNIVERSIDADE ESTADUAL PAULISTA JÚLIO DE MESQUITA FILHO
FACULDADE DE CIÊNCIAS
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Trabalho de Conclusão de Curso

Autor
Lawrence Donizetti Custódio Faccine Júnior

Aplicação Web para Monitoramento e Análise de Rede

Bauru, 2019

UNIVERSIDADE ESTADUAL PAULISTA JÚLIO DE MESQUITA FILHO
FACULDADE DE CIÊNCIAS
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Trabalho de Conclusão de Curso

Autor
Lawrence Donizetti Custódio Faccine Júnior

Aplicação Web para Monitoramento e Análise de Rede

Trabalho de conclusão de curso apresentado ao curso de Bacharelado em Sistemas de Informação da Universidade Estadual Paulista "Júlio de Mesquita Filho" como exigência parcial para obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Kelton Augusto Pontara da Costa

Bauru, 2019

Resumo

Esta monografia visa o desenvolvimento de uma aplicação Web para auxiliar no gerenciamento e análise de rede, através da exibição de gráficos em tempo real utilizando o protocolo SNMP (Protocolo Simples de gerenciamento de redes). O problema a ser resolvido trata-se da falta de uma ferramenta gratuita que possa ser acessível de qualquer localidade por um administrador de redes e auxiliá-lo. Sendo assim, foi desenvolvida a aplicação web para gerenciamento e análise de rede, sendo uma opção de ferramenta gratuita em dar suporte ao gerenciador de redes, visto a necessidade encontrada.

Palavras-chave: Análise de Redes. SNMP. Web.

Abstract

This monograph aims to develop a web application to assist in network management and analysis by displaying graphs in real time using the Simple Network Management Protocol (SNMP). The problem to be solved is the lack of a free tool that can be accessed from anywhere by a network administrator and assisting him. Thus, the web application for network management and analysis was developed, being a free tool option to support the network manager, given the need found.

Keywords: Network analysis. SNMP. Web.

Lista de Figuras

Lista de Quadros

Lista de Siglas

CSS	Cascading Style Sheets
HTML	Linguagem de Marcação de HiperTexto
IP	Internet Protocol
MIB	Management Information Base
MRTG	Multi Router Traffic Grapher
OID	Object Identifier
SNMP	Simple Network Management Protocol

Sumário

1 Introdução

Nos dias atuais, com o avanço da tecnologia, muitos ambientes públicos e privados adotam utilização de uma rede de internet para seus usuários, como WI-FI em praças, shoppings, hotéis, restaurantes ou como na maioria das empresas, já que hoje em dia é imprescindível que empresas possuam internet para deixar seus processos mais rápidos e ágeis. Devido ao alto número de usuários que essas redes podem ter, o administrador de redes necessita de ferramentas que o auxiliem no gerenciamento da mesma. Tais ferramentas já são existentes, porém, não foi encontrada nenhuma que possa ser acessível em uma aplicação Web (não dependendo de um software a ser instalado).

Portanto foi desenvolvida uma ferramenta Web que através do protocolo SNMP (Simple Network Management Protocol), analisará a rede em que o usuário está conectado, armazenando as informações coletadas em um banco de dados para posterior exibição de gráficos em tempo real, assim, o usuário poderá tomar decisões sobre a rede a qual ele gerencia. O objetivo desta ferramenta não é a segurança da rede, sendo esta, responsabilidade da tomada de decisão do administrador. Por se tratar de uma aplicação Web, o usuário terá a condição de acessar esta ferramenta gratuitamente por qualquer computador ou notebook, através de um endereço web.

2 Detalhamento do Problema

Ao ser analisado que ferramentas de gerenciamento e análise de rede são essenciais para confiabilidade e qualidade da conexão e que, a maioria das ferramentas atuais são softwares pagos e precisam ser instalados na máquina pessoal do usuário bem como não foram descobertas soluções acessíveis a partir da web, foi encontrada a necessidade de desenvolver uma plataforma Web online gratuita em que seja possível acessá-la de qualquer computador. Além disso, foi verificada a necessidade de fornecer ao usuário a amostra de dados em tempo real, para que a tomada de decisões do mesmo seja eficiente e ágil.

Quando uma empresa possui uma fraca administração de rede, sua credibilidade é duvidosa, pois a sua conexão será provavelmente ruim, lenta, e poucos usuários conseguirão utilizá-la, afastando com isso, possíveis clientes, consumidores, afetando a produtividade de seus funcionários e até deixando brechas para possíveis ameaças de rede. Sendo assim, impacta financeiramente essa empresa que possui esse tipo de rede precária, que perde diante da concorrência, por exemplo, em produtividade.

Para evitar os problemas citados acima, a empresa terá que possuir um administrador de redes capacitado, porém, os softwares para auxiliá-lo são em sua maioria pagos e depende do investimento que a empresa realizar. Com a ferramenta desenvolvida, a empresa poderá auxiliar o administrador de redes de forma gratuita, para este profissional manter a rede estável e segura, ou seja, para que não aconteça lentidão na rede e os usuários não tenham suas informações roubadas por alguém mal intencionado.

São necessários alguns requisitos para que a ferramenta seja utilizada corretamente pelo administrador de redes. Os requisitos funcionais para o sistema são:

- Estar conectado com a rede a ser analisada;
- Ativar o protocolo SNMP para análise dos dados;
- Possuir acesso à internet estável para conectar a ferramenta; e
- Ter um computador pessoal para acessar a ferramenta;

Não há custos sobre a ferramenta, pois a mesma será gratuita.

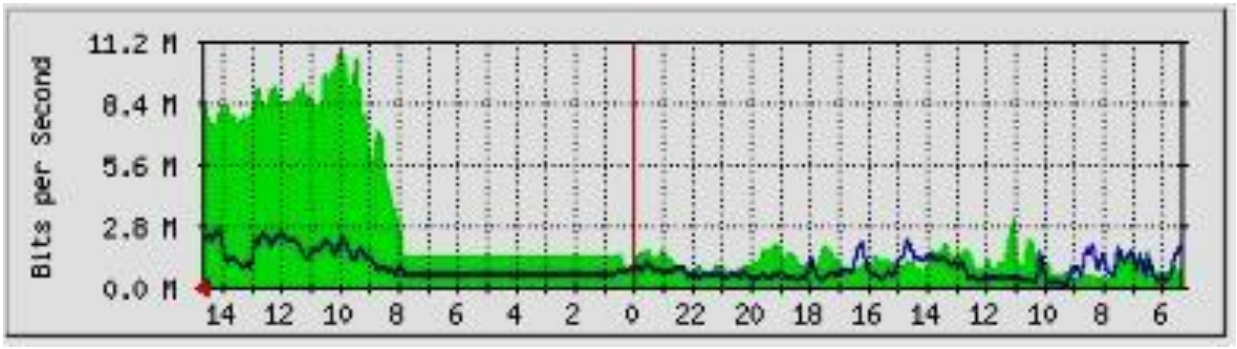
3 Soluções Existentes

Hoje em dia, existem várias ferramentas de análise de tráfego para auxiliar o administrador a gerenciar sua rede e ajudá-lo a tomar medidas e precauções com base nas informações apresentadas pela ferramenta, não são todas as ferramentas que utilizam o protocolo SNMP. Neste capítulo, será abordada a ferramenta MRTG, que utiliza o protocolo SNMP para coletar os dados do tráfego nos equipamentos de rede e com esses dados, gerar gráficos para análise e gerenciamento.

Multi Router Traffic Grapher (MRTG) é um programa escrito na linguagem Perl, gratuito e de distribuição aberta que utilizando o protocolo SNMP, ele obtém acesso às variáveis de tráfego dos dispositivos gerenciados e posteriormente constrói gráficos com as informações coletadas.

Os gráficos são gerados em páginas HTML e podem ser de quatro tipos: diário, semanal, mensal e anual. O gráfico diário, mostrada na Figura 1, é atualizado a cada 5 minutos e possui abscissa com aproximadamente 33 horas. Além dos gráficos, são também fornecidas informações textuais de valor atual, valor médio, valor máximo, percentual, entre outras.

Figura 1: Gráfico Diário do MRTG



Fonte: DIAS, Beethovem Z.; JUNIOR, Nilton A. (2001).

Para o funcionamento do MRTG, além de ter que instalar a própria ferramenta, é necessário também à instalação do Perl. Isso é um diferencial para a ferramenta desenvolvida, já que ele não necessitará de instalação e o único pré-requisito será que o usuário possua um navegador instalado.

4 Tecnologias Utilizadas

Neste capítulo estão apresentadas as tecnologias que foram utilizadas para o desenvolvimento da ferramenta.

4.1 Front-End

Para o desenvolvimento do Front-End será utilizado o trio HTML, CSS e Javascript, o que tornará mais fácil o desenvolvimento da página Web.

4.1.1 Html

HTML (Linguagem de Marcação de HiperTexto) é o componente mais básico da web. Ela serve para definir o conteúdo e a estrutura básica de uma página web. HyperText (HiperTexto) refere-se aos links (ligações) que conectam uma página a outra, seja dentro de um mesmo website ou entre websites diferentes. A HTML usa Markup (Marcação) para mostrar textos, imagens e outros conteúdos para visualização em um navegador Internet. Marcadores HTML incluem elementos especiais, tais como <head>, <title>, <body>, <header>, <footer>, <article>, <section>, <p>, , e <img>, entre outros (MOZILLA, 2019).

4.1.2 Css

O Cascading Style Sheets (CSS) é uma linguagem utilizada para definir a apresentação (aparência) de documentos que adotam para o seu desenvolvimento linguagens de marcação (como XML, HTML e XHTML e etc..). O CSS define como serão exibidos os elementos contidos no código de um documento e sua maior vantagem é efetuar a separação entre o formato e o conteúdo de um documento. Sendo assim, o CSS encarregou-se da aplicação dos estilos necessários para a aparência dela. Isto é feito por meio da criação de um arquivo externo que contém todas as regras aplicadas e com isto, é possível fazer alterações de estilo em todas as páginas de um site e outros documentos que utilizam CSS de forma fácil e rápida.

4.1.3 JavaScript

JavaScript é uma linguagem de programação client-side. Ela é utilizada para controlar o HTML e o CSS para manipular comportamentos na página. Criada por Brendan Eich na Netscape (um dos precursores dos navegadores web), era chamada de LiveScript, mas logo seu nome foi mudado para JavaScript. Mesmo assim o nome original é ECMAScript, por que o JavaScript é mantido pela European Computer Manufacturer's Association (MOZILLA, 2019).

4.2 Back-End

Para o lado do servidor, foi utilizado Python por ele ser uma linguagem simples e de fácil aprendizado. Ele também possui uma boa comunidade com muitas informações já disponibilizadas que ajudaram no desenvolvimento da ferramenta, além de ser uma ótima linguagem para aplicações em tempo real, o que se enquadra com a ferramenta desenvolvida.

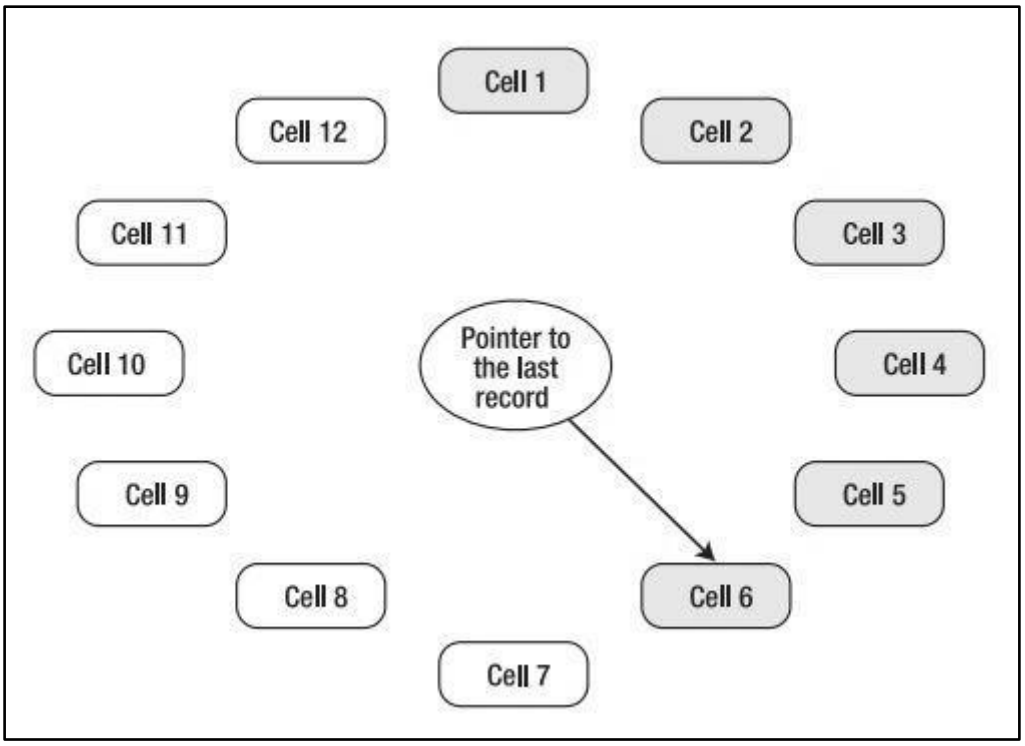
4.2.1 CherryPy

Cherrypy é um framework gratuito, disponível para a linguagem Python, que é utilizado para desenvolvimento web sendo que, criar aplicativos usando ele é muito semelhante a criar outros programas orientados a objetos usando Python, o que facilita o aprendizado e a utilização do framework (CherryPy, 2019). Por ele seguir as convenções do Python, muitos dos módulos disponíveis para a linguagem podem ser utilizados junto ao framework, além de propriamente fornecer ferramentas e plugins para serem usados no desenvolvimento. O Cherrypy inclui seu próprio servidor HTTP pronto para produção para hospedar o aplicativo, mas ele também pode ser implantado em qualquer gateway compatível com WSGI.

4.2.2 RRDtool

O RRDtool (Round-Robin Database tool) é um banco de dados desenvolvido por Tobias Oetiker que visa lidar com dados de séries temporais, como largura de banda da rede, temperaturas ou carga da CPU. Os dados são armazenados de forma circular como um buffer, similar ao mostrado na Figura 2, sendo assim o espaço ocupado pelo armazenamento não aumenta com o tempo, permanecendo constante, mas quando chega ao limite ele começa a sobrescrever as células mais antigas.

Figura 2: Estrutura do RRDtool



Fonte: SILEIKA (2014).

Além da capacidade de armazenamento, o RRDtool conta também com um módulo de geração de gráficos com os dados que foram previamente armazenados em seu banco. Como o foco desse banco são dados de séries temporais como, por exemplo, os dados coletados de uma rede, e ainda consegue gerar os próprios gráficos com os dados contidos em seu banco, ele se torna uma ótima solução para ser aplicada, sendo que o RRDtool possui suporte para Python, que foi a linguagem utilizada para o desenvolvimento desta ferramenta.

4.2.3 PySNMP

PySNMP é uma biblioteca para Python gratuita e de código aberto que possui todos os funcionamentos do protocolo SNMP podendo atuar em funções como agente, gerente ou proxy. Ela será utilizada pois, feito uma pesquisa rápida entre as outras bibliotecas, o resultado foi que o PySNMP é a mais completa e a que tem melhor documentação, contendo exemplos práticos de utilização.

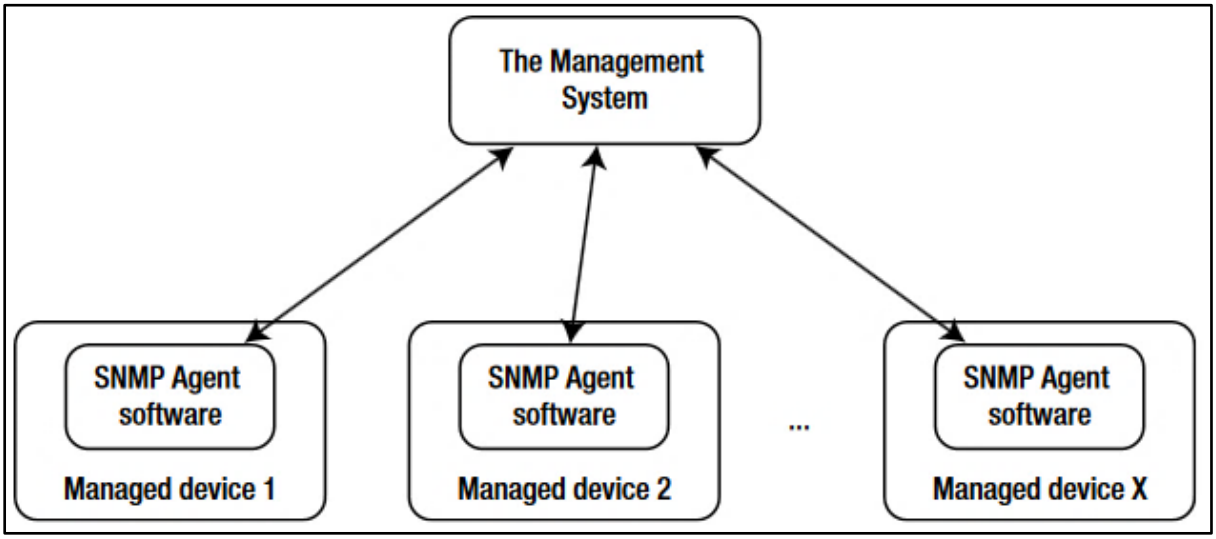
4.3 Protocolo SNMP

SNMP, Simple Network Management Protocol, é um protocolo, especificado na RFC 1157, que foi criado para facilitar o monitoramento e o gerenciamento de redes. Ele acompanha o status da rede em tempo real de forma simples e fácil.

Basicamente ele requisita informações, através de uma aplicação de gerenciamento, para dispositivos que estão conectados a rede, como por exemplo computadores Desktop, que enviam essas informações de volta para que a aplicação de gerenciamento as use, gerando gráficos para monitoramento por exemplo (DIAS; JUNIOR, 2002). Sendo o gerenciamento de rede o foco da ferramenta, foi utilizado esse protocolo para a coleta de informações.

Segundo Sileika (2014) e conforme mostrado na Figura 3, existem três componentes em um sistema gerenciado pelo SNMP: o gerenciador do sistema (Management System), que é responsável por gerenciar os dispositivos; os dispositivos gerenciados (Managed devices); e o agente SNMP (SNMP Agent software), que é uma aplicação que executa em cada um dos dispositivos gerenciados e interage com o gerenciador.

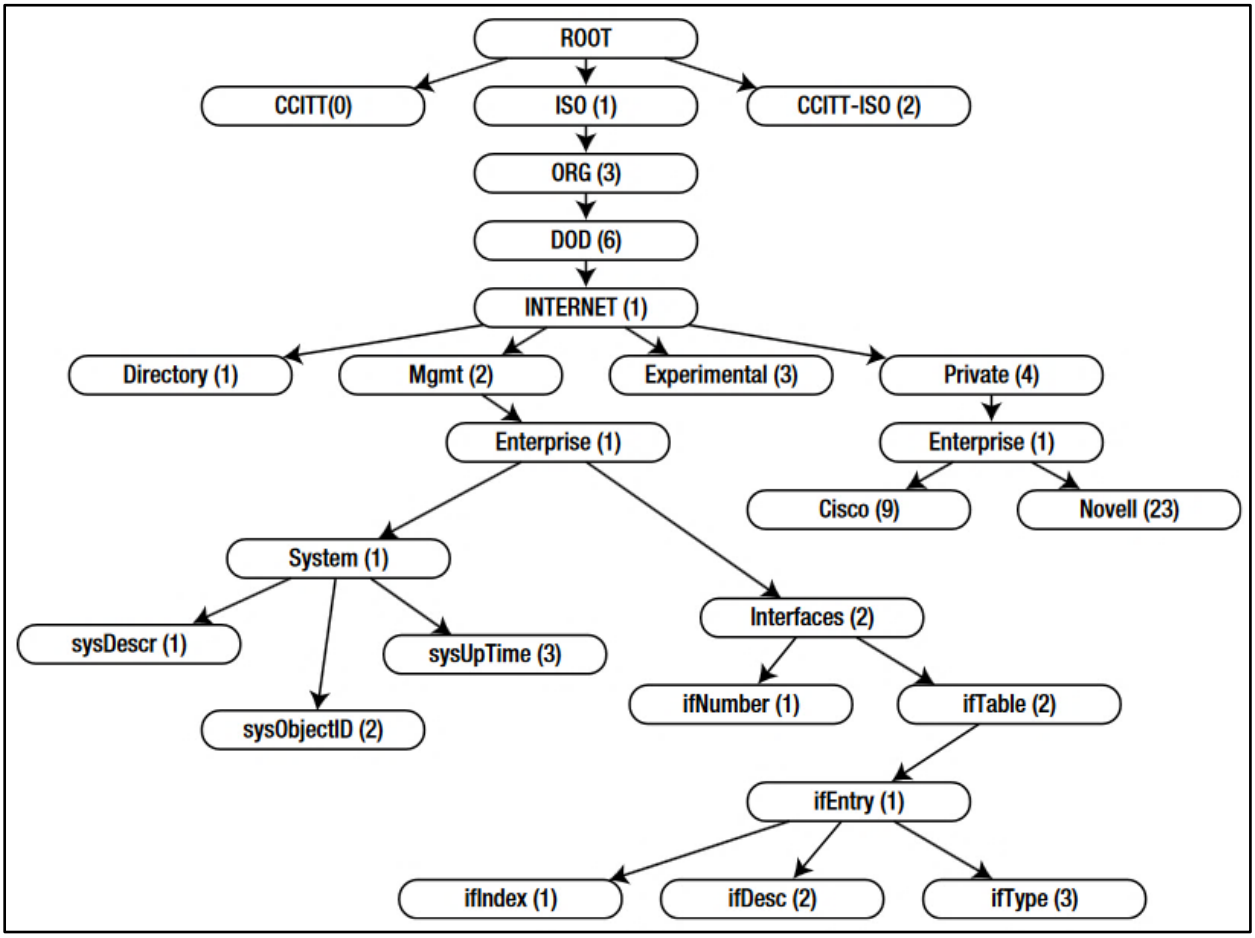
Figura 3: Componentes do SNMP



Fonte: SILEIKA (2014).

"A Base de Informações de Gerenciamento (Management Information Base - MIB) pode ser imaginada como um banco virtual de informações que guarda objetos gerenciados cujos valores, refletem o "estado" atual da rede." (KUROSE; ROSS, 2014, p. 566). Esses valores podem ser requisitados pelo gerenciador para verificar o funcionamento desse sistema gerenciado. Cada dispositivo possui uma MIB associada que descreve a estrutura dos dados de gerenciamento daquele sistema. Para identificar cada uma das informações diferentes que estão dentro dessa MIB, existem os Objects Identifiers (OIDs). Cada OID é representada por um nó na árvore de objetos como mostrado na Figura 4, mas ele contém todos os identificadores numéricos que levam até o OID atual, como por exemplo a OID que foi designada a organização Cisco seria a 1.3.6.1.4.1.9 (SILEIKA, 2014).

Figura 4: Árvore de OIDs



Fonte: SILEIKA (2014).

Para a coleta de dados entre o gerente e o dispositivo gerenciado, existem três operações que podem ser efetuadas:

1. GetRequest: é feita uma requisição para o recuperar informação de uma ou um conjunto de OIDs;
2. GetNextRequest: é feita uma requisição para recuperar todas as informações que estão abaixo do nó OID que foi requisitado; e
3. GetBulkRequest: coleta uma grande quantidade de informações como se fossem feitas várias requisições do tipo GetNextRequest.

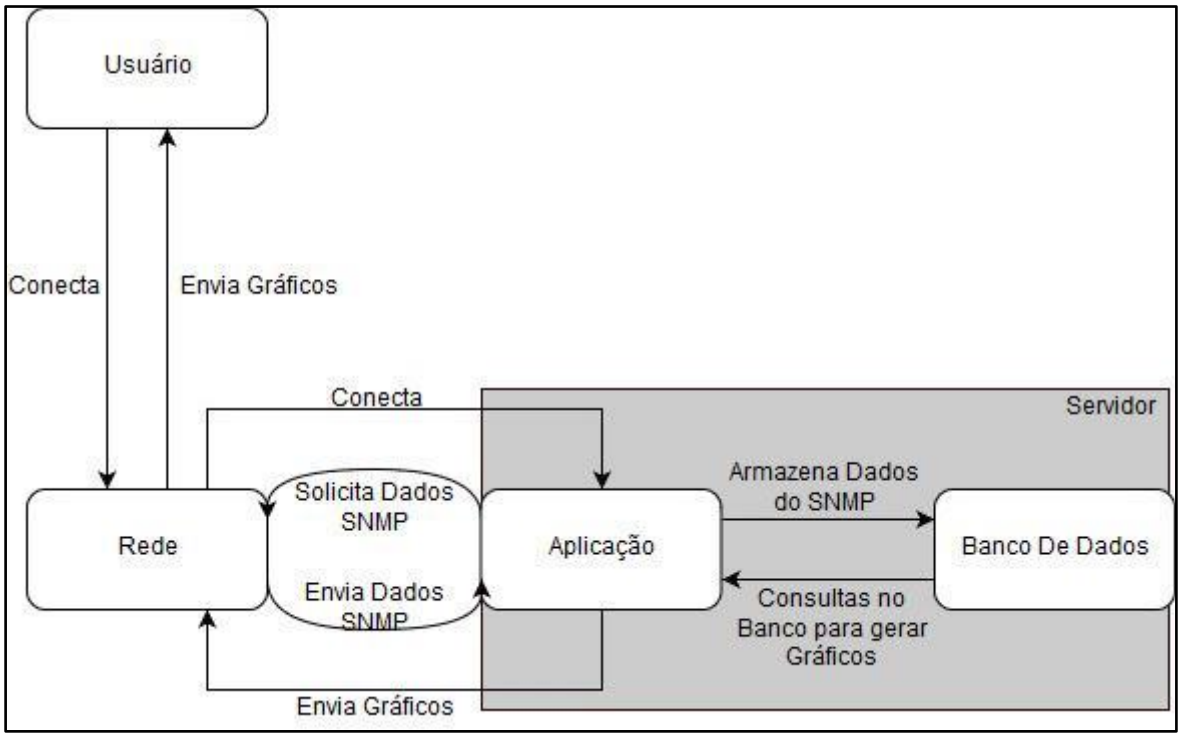
Na versão 2 do protocolo SNMP, é definido duas *“community strings”* que servem para a autenticação do gerente antes que ele possa fazer alguma coleta ou manipulação dos dados, é como se fosse uma chave de segurança que foi definida pelo gerente de rede ou o fabricante do dispositivo. Uma dessas *“community strings”* é definida como “read-only”, assim o gerente só poderá ler os dados contidos nas

MIB, diferente da “read/write”, que significa que o gerente pode ler e escrever, alterar, as informações contidas na MIB do dispositivo gerenciado. Como a ferramenta só trabalha com informações do tipo “read-only”, sendo o foco o monitoramento, não é necessário fazer alterações nas informações contidas na MIB.

5 Especificação da Ferramenta

Neste capítulo, está apresentada uma descrição geral da ferramenta, bem como suas funcionalidades e como é sua interação com o usuário. Na Figura 5, se encontra um esquema geral de como funcionará a ferramenta quando posta em prática.

Figura 5: Esquema Geral de Funcionamento



Fonte: Elaborado pelo autor.

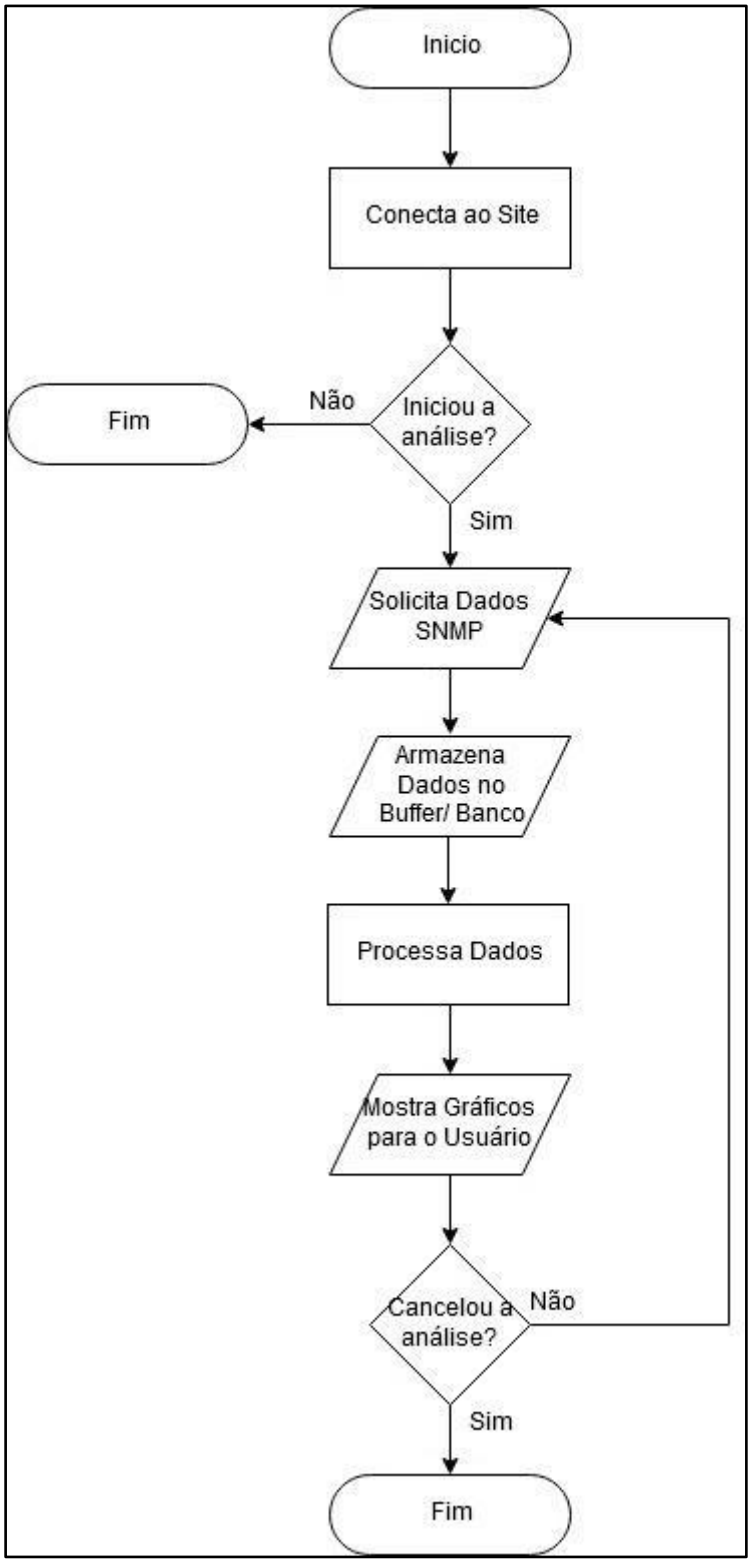
A aplicação possui quatro pilares principais para seu funcionamento, sendo eles: o usuário, a rede, a aplicação e o banco de dados. A aplicação funcionará com duas threads principais para que ela funcione em tempo real, pois a medida que ela coleta os dados necessários, já serão plotados gráficos para o administrador de rede fazer o gerenciamento dessa rede.

Para que a aplicação comece com a coleta de dados, o usuário precisa antes se conectar ao site, já que essa é uma aplicação *Web Based*, utilizando a rede que será posteriormente utilizada para a coleta dos dados.

Com a conexão feita, a aplicação reconhecerá a rede que o usuário está conectado e solicitará a coleta de dados do protocolo SNMP. A seguir, a aplicação

irá receber os dados coletados dos dispositivos da rede, armazenando-os dentro do banco de dados para que eles possam ser armazenados e posteriormente utilizados para gerar os gráficos necessários para o usuário poder fazer o gerenciamento da rede analisada e tomar medidas preventivas para manter uma rede com bom funcionamento. Para facilitar mais o entendimento de como a ferramenta funcionará, a Figura 6 traz um fluxograma dos eventos que irão acontecer no decorrer da execução do programa.

Figura 6: Fluxograma da Ferramenta



Fonte: Elaborado pelo autor.

5.1 Diagrama de Casos de Uso

Um diagrama de casos de uso é bastante utilizado em engenharia de software para demonstrar as interações dos atores com o sistema. Ele ajuda a representar os cenários que o sistema vai interagir com os usuários ou sistemas externos e o escopo do sistema a ser projetado (LUCIDCHART).

A aplicação pode ser representado em um caso de uso bem simples como mostra a Quadro 1, com um fluxo principal e só 2 fluxos de exceção.

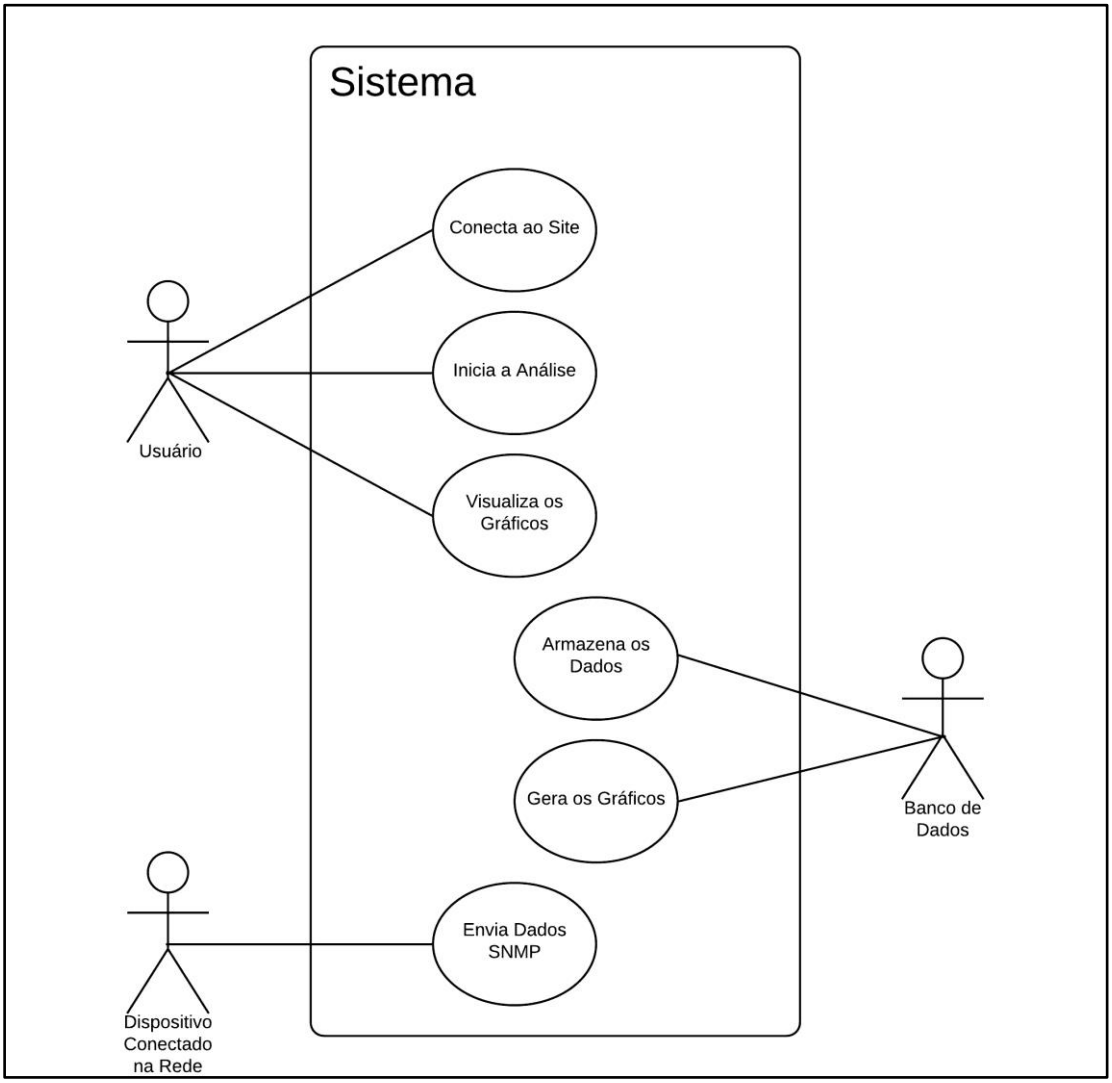
Quadro 1: Caso de Uso

Sumário: Usuário Visualiza Gráficos sobre informações de sua rede.
Ator Principal: Usuário.
Fluxo Principal: 1) Usuário se conecta ao site. 2) Usuário inicia a análise. 3) Sistema solicita dados SNMP ao dispositivo conectado na rede. 4) Sistema armazena esses dados no Banco. 5) Sistema pega os dados do banco e os processa. 6) Sistema envia gráficos ao usuário. 7) Repetição dos passos 3 ao 6 até que o usuário cancele a análise.
Fluxo de Exceção (2): Usuário não inicia a análise. a) Sistema não tem dados para gerar gráficos
Fluxo de Exceção (7): Usuário cancela a análise. a) Sistema para de solicitar dados. b) Gráficos não são atualizados.

Fonte: Elaborado pelo autor.

A Figura 7 traz o diagrama de caso de uso da ferramenta. Ele contém três atores principais sendo eles o Usuário, o Dispositivo conectado na rede e o Banco de dados. O primeiro ator, o usuário, ficará com a função de se conectar ao site e dar início a análise e captura dos dados fazendo com que o dispositivo que está na rede, segundo ator, envie os dados do protocolo para o sistema. O banco de dados ficará responsável pelo armazenamento dos dados que estão chegando do dispositivo para que posteriormente sejam gerados os gráficos para o usuário visualizar.

Figura 7: Diagrama de Casos de Uso



Fonte: Elaborado pelo autor.

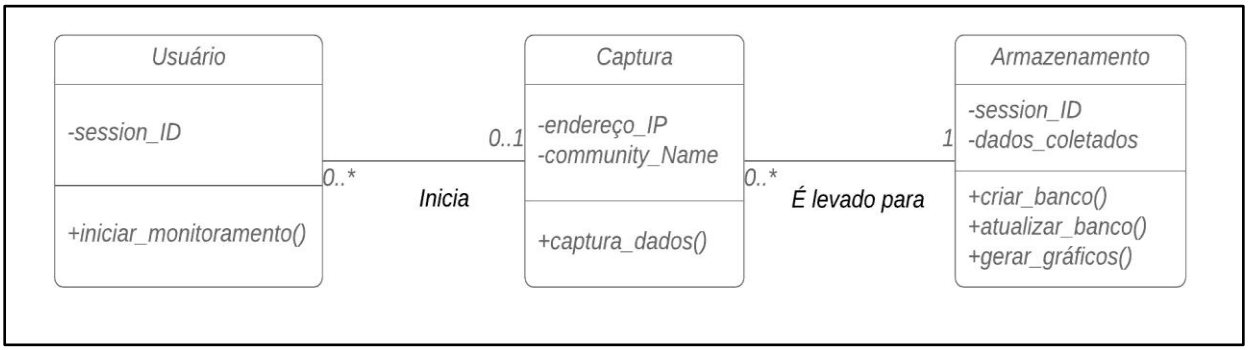
5.2 Diagrama de Classes

Diagrama de Classes estão entre os mais úteis tipos de diagramas UML. Eles mapeiam a estrutura do sistema modelando suas classes, atributos e operação além de mostrar as relações entre objetos (LUCIDCHART).

O diagrama de classes da aplicação possui a princípio três classes básicas: Usuário, Captura, Armazenamento, como mostra na Figura 8. A primeira será utilizada para receber a solicitação do usuário e iniciar a análise. A ‘Captura’ irá capturar os dados provenientes do dispositivo da rede. A terceira classe, ‘Armazenamento’, ficará responsável pelo armazenamento dos dados no banco de

dados e, como o RRDtool consegue gerar os próprios gráficos com os dados contidos nele, também ficara responsável por gerar os gráficos para posteriormente serem mostrados para o usuário.

Figura 8: Diagrama de Classes



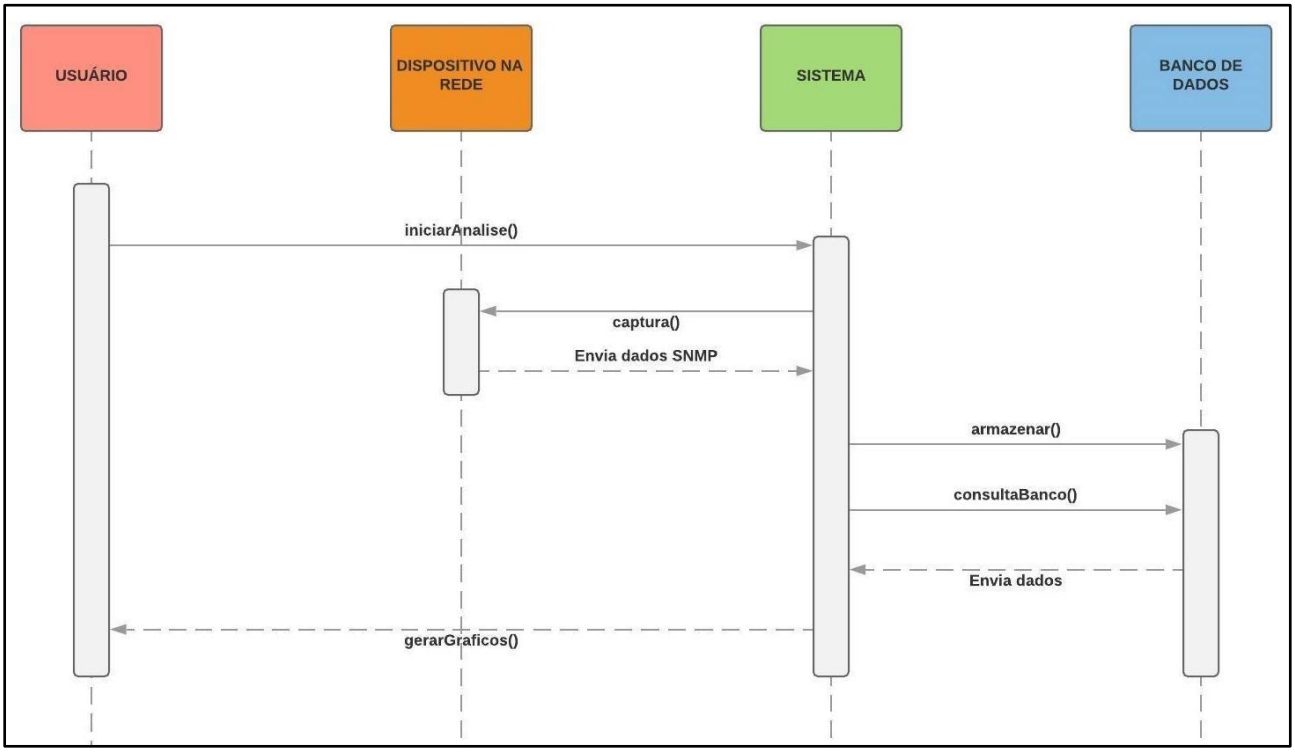
Fonte: Elaborado pelo autor.

5.3 Diagrama de Sequência UML

O diagrama de sequência é usado para se visualizar as linhas de vida dos objetos e processos, bem como as mensagens que são trocadas entre eles para desempenhar alguma função antes do término da linha de vida (LUCIDCHART).

A maior parte da troca de mensagens do sistema ficará sendo entre o dispositivo na rede, o sistema e o banco de dados, como mostrado na Figura 9, sendo que o dispositivo fica mandando os dados para o sistema, via requisição, que os armazena no banco de dados. A medida que os dados vão sendo colocados no banco de dados, o sistema irá fazer consultas no banco para trazer os dados necessários para gerar os gráficos na tela do usuário.

Figura 9: Diagrama de Sequência UML



Fonte: Elaborado pelo autor.

5.4 Utilização do Protocolo SNMP

Para o desenvolvimento da ferramenta, foi utilizado o protocolo em sua versão 2 e, como o protocolo tem um número grande de OIDs suportadas e algumas são específicas para determinados equipamentos, para a aplicação foi escolhida algumas que possuem relação com a rede, como por exemplos as que mostram taxas de entrada e saída de bytes, assim as OIDs escolhidas estão listadas no Quadro 2. Para a coleta de dados, a operação escolhida foi a GetRequest pois as OIDs que seriam utilizadas já estavam definidas e não seria necessário a coleta de mais informações, e também para que o arquivo do banco de dados não ficasse muito grande sobrecarregando o servidor.

Quadro 2: OIDs utilizadas na ferramenta

MIB	Nome	OID	Descrição
IF-MIB	ifInOctets	1.3.6.1.2.1.2.2.1.10	Taxa de bytes recebidos.
IF-MIB	ifOutOctets	1.3.6.1.2.1.2.2.1.16	Taxa de bytes enviados.
IF-MIB	ifInDiscards	1.3.6.1.2.1.2.2.1.13	Taxa de descartes de pacotes de entrada.
IF-MIB	ifOutDiscards	1.3.6.1.2.1.2.2.1.19	Taxa de descartes de pacotes de saída.
IF-MIB	ifInErrors	1.3.6.1.2.1.2.2.1.14	Taxa de pacotes com erros de entrada.
IF-MIB	ifOutErrors	1.3.6.1.2.1.2.2.1.20	Taxa de pacotes com erros de saída.
IF-MIB	ifInUnknownProtos	1.3.6.1.2.1.2.2.1.15	Taxa de pacotes de protocolos desconhecidos recebidos.
TCP-MIB	tcpAttemptFails	1.3.6.1.2.1.6.7	Número de tentativas de conexão falhadas.
TCP-MIB	tcpInErrs	1.3.6.1.2.1.6.14	Número de pacotes recebidos com erro.
TCP-MIB	tcpInSegs	1.3.6.1.2.1.6.10	Taxa de segmentos TCP recebidos.
TCP-MIB	tcpOutSegs	1.3.6.1.2.1.6.11	Taxa de segmentos TCP enviados.
UDP-MIB	udpInDatagrams	1.3.6.1.2.1.7.1	Número total de datagramas UDP recebidos.
UDP-MIB	udpOutDatagrams	1.3.6.1.2.1.7.4	Número total de datagramas UDP enviados.
UDP-MIB	udpNoPorts	1.3.6.1.2.1.7.2	Taxa de datagramas que não foram enviados para uma porta válida.

Fonte: Elaborado pelo autor.

Cada uma dessas OIDS trazem informações diferentes, com o ifInOctets e ifOutOctets, é possível ver a taxa de bytes recebidos e enviados da interface que está sendo monitorada. O ifOutOctets e o ifOutDiscards podem sinalizar o congestionamento da rede caso o segundo tenha um número alto de descartes mas

o primeiro um número baixo de bytes de saída. As OIDS ifInErrors e ifOutErrors mostram quantos pacotes chegaram/saíram e que continham erros. Com o ifInUnknownProtos, é mostrado a taxa de pacotes desconhecidos recebidos e que foram descartados, portanto ele e o ifInDiscarts crescerão proporcionalmente.

Verificando o tcpAttemptFails, pode-se medir a confiabilidade da rede, sendo que uma rede mais confiável terá menos falhas, com o tcpInErrs, é mostrado o número de segmentos recebidos que continham erros e, as OIDS tcpInSegs e tcpOutSegs mostram a taxa de segmentos TCP que entram e saem da interface.

Por fim, tem-se os udpInDatagrams e udpOutDatagrams que mostram quantos datagramas UDP foram recebidos e enviados pela interface e o udpNoPorts nos mostra quando a interface está recebendo datagramas de uma aplicação inválida, sendo que um número alto pode resultar em problemas de performance.

5.5 Banco de Dados RRDtool

Como dito anteriormente, o banco de dados utilizado foi o RRDtool, portanto neste capítulo está apresentado como que o banco foi utilizado na ferramenta, contendo alguns trechos de código implementando a biblioteca para Python do banco de dados.

A criação do banco foi feita utilizando o código mostrado na Figura 10, rrdtool.create é a função de criação do banco e em seguida é o nome que vai ser atribuído a ele. Para cada acesso da página será gerado um arquivo do banco com as informações da rede. Os próximos comandos ‘--start’ e ‘--step’, significam quando o banco irá iniciar, nesse caso ‘N’ significa o “agora”, e o período que o banco será atualizado em segundos. Cada “DS” (Data Source) significa uma fonte de dados, onde cada um é uma OID que foi coletada para monitoramento. Em seguida vem o DST (Data Source Type) que mostra o tipo do dado que está sendo coletado, como os dados de rede do protocolo SNMP nunca diminuem, só incrementam, o tipo atribuído foi “COUNTER”. Os próximos argumentos são o *hertbeat*, que é o tempo máximo em segundos que podem passar entre as atualizações do banco em que não tenham nenhuma atualização concreta, e o mínimo e máximo aceitos no banco, que no exemplo foram colocados ‘0’ de limite inferior e ‘U’ que significa “sem limites”. O RRA (Round Robin Archieve) é como os dados vão ser armazenados no

banco, nesse caso será armazenado a média dos dados coletados (Average). Em seguida temos o ‘0.5’ que significa a porcentagem dos dados que podem ser *“Unknown”*, ou seja, desconhecida e ainda assim ser armazenada no banco. O número '1’ indica quantos dos dados coletados serão usados para consolidar as informações do banco e o último número indica quantos resultados para cada ‘DS’ poderão ser armazenados no banco.

Figura 10: Função de criação do RDDtool em Python

```
rrdtool.create('%s/%s.rrd' % (temp_dir, id),
                    '--start', 'N',
                    '--step', '1',
                    'DS:ifInOctets:COUNTER:60:0:U',
                    'DS:ifOutOctets:COUNTER:60:0:U',
                    'DS:ifInDiscards:COUNTER:60:0:U',
                    'DS:ifOutDiscards:COUNTER:60:0:U',
                    'DS:ifInErrors:COUNTER:60:0:U',
                    'DS:ifOutErrors:COUNTER:60:0:U',
                    'DS:ifInUnknownProtos:COUNTER:60:0:U',
                    'DS:tcpAttemptFails:COUNTER:60:0:U',
                    'DS:tcpInErrs:COUNTER:60:0:U',
                    'DS:tcpInSegs:COUNTER:60:0:U',
                    'DS:tcpOutSegs:COUNTER:60:0:U',
                    'DS:udpInDatagrams:COUNTER:60:0:U',
                    'DS:udpOutDatagrams:COUNTER:60:0:U',
                    'DS:udpNoPorts:COUNTER:60:0:U',
                    'RRA:AVERAGE:0.5:1:90000')
```

Fonte: Elaborado pelo autor.

Para atualizar o banco com novos dados, foi utilizada a função mostrada na Figura 11, o ‘rrdtool.update’. Basicamente os argumentos são o nome do arquivo do banco que vai ser atualizado, bem como o diretório em que este banco se encontra, quais informações estão sendo atualizadas e o momento em que foi atualizado, no caso ‘N’ de “agora".

Figura 11: Função de update do RRdtool em Python

```
rrdtool.update('%s/%s.rrd' % (temp_dir, id),
                   'N:%s:%s:%s:%s:%s:%s:%s:%s:%s:%s:%s:%s:%s' %
                    (ifInOctets, ifOutOctets, ifInDiscards,
                    ifOutDiscards, ifInErrors, ifOutErrors,
                    ifInUnknownProtos, tcpAttemptFails,
                    tcpInErrs, tcpInSegs, tcpOutSegs,
                    udpInDatagrams, udpOutDatagrams,
                    udpNoPorts))

```

Fonte: Elaborado pelo autor.

A Figura 12 mostra uma das funções que foi utilizada para gerar os gráficos de monitoramento da ferramenta, ‘rrdtool.graph’ é o nome da função seguida do diretório em que esse gráfico será armazenado, o formato da imagem, o período em que se inicia o gráfico e os limites superiores e inferiores. Em seguida, no ‘DEF’ é definido quais são os dados que serão mostrados no gráfico, a localização do arquivo em que estão esses dados e o ‘RRA’ que foi definido. O comando ‘GPRINT’ coloca na parte inferior do gráfico uma informação adicional, ele mostra o valor máximo do dado que foi armazenado no banco até aquele período. O ultimo comando, ‘LINE2’ cria uma legenda para o gráfico na parte inferior da imagem.

Figura 12: Função de gerar gráficos do RRDtool em Python

```
rrdtool.graph('%s/IfOctets.png' % (temp_dir),
              '--imgformat', 'PNG',
              '--title', 'Média de Entrada e Saída de Bytes',
              '--start', '-300',
              '--upper-limit', '100000',
              '--lower-limit', '-50000',
              'DEF:ifInOctets=%s/%s.rrd:ifInOctets:AVERAGE' % (temp_dir, id),
              'DEF:ifOutOctets=%s/%s.rrd:ifOutOctets:AVERAGE' % (temp_dir, id),
              'GPRINT:ifInOctets:MAX:Max ifInOctets\: %8.2lf',
              'LINE2:ifInOctets#ff0000:ifInOctets',
              'GPRINT:ifOutOctets:MAX:Max ifOutOctets\: %8.2lf',
              'LINE2:ifOutOctets#0000ff:ifOutOctets'
              )

```

Fonte: Elaborado pelo autor.

6 Demonstração da Ferramenta

Neste capítulo será demonstrado o funcionamento da ferramenta desenvolvida, como utilizá-la, como instalar suas dependências e alguns casos de teste que foram feitos utilizando a ferramenta.

6.1 Instalação da Ferramenta

Uma breve explicação de como instalar a ferramenta, bem como os comandos utilizados está apresentada neste capítulo.

A ferramenta foi desenvolvida no sistema operacional Ubuntu, e devido aos requerimentos, tal como a instalação do banco de dados RRDtool, é preferível que a utilização da ferramenta seja em ambiente Linux. Neste capítulo somente foi mostrado os comandos para instalação dos requisitos no ambiente Ubuntu.

A linguagem principal usada na aplicação foi o Python na versão 3, o comando de instalação está descrito na Figura 13.

Figura 13: Comando de instalação do Python

```
1  $ sudo apt-get install python3
2
```

Fonte: Python (2019).

O Python possui seu próprio gerenciador de pacotes, o Pip, o que facilita na hora de instalar as bibliotecas utilizadas. Conforme mostrado na Figura 14, a instalação do Pip é feita através desse comando.

Figura 14: Comando de instalação do Pip

```
1  $ sudo apt-get install python3-pip
2
```

Fonte: Python (2019).

A ferramenta também utiliza do framework web CherryPy na sua versão 18.4.0, a instalação é feita conforme mostrado na Figura 15.

Figura 15: Comando de instalação do CherryPy

```
1  $ pip3 install cherrypy
2
```

Fonte: Fonte: CherryPy (2019).

Para a coleta de dados do protocolo SNMP, a biblioteca utilizada foi o PySNMP. Para o funcionamento perfeito ele necessita de algumas dependências, como a instalação das MIBs, já outras dependências são instaladas junto com a biblioteca. Para instalar a biblioteca utiliza-se o comando mostrado a seguir na Figura 16.

Figura 16: Comando de instalação do PySNMP

```
1  $ pip3 install pysnmp-mibs
2  $ pip3 install pysnmp
3
```

Fonte: PySNMP (2019)

Para a instalação do banco de dados, é necessário instalar suas dependências antes, visto que não é instalado tudo junto como no caso da biblioteca PySNMP. Os comandos para instalar as dependências do RRDtool para Python, e instalar a biblioteca Python do RRDtool é mostrado na Figura 17.

Figura 17: Comando de instalação do RRDtool

```
1  $ sudo apt-get install librrd-dev libpython3-dev
2  $ pip3 install rrdtool
3
```

Fonte: Python-RRDtool (2016)

Com isso, todas as dependências foram instaladas, sobrando apenas inicializar o servidor do CherryPy para o funcionamento da ferramenta. Para iniciar o servidor, no diretório em que se encontra o ‘app.py’, que é o arquivo que contém o código do servidor, utilizar o comando mostrado na Figura 18, assim, inicializando a ferramenta.

Figura 18: Comando de inicialização do CherryPy

```
$ python3 app.py
```

Fonte: CherryPy (2019).

6.2 Utilização da Ferramenta

A página inicial, como mostrado na Figura 19, contém uma breve explicação sobre a aplicação e mostra um quadro, o mesmo apresentado anteriormente, contendo as OIDs utilizadas para a geração dos gráficos de monitoramento, bem como suas respectivas descrições. Para iniciar a análise, o usuário tem que colocar o endereço IP (Internet Protocol) do alvo do monitoramento, bem como o *"community name"* que foi estipulado para o dispositivo alvo (por padrão alguns equipamentos já vêm com o "*community name"* definido como "public"). Preenchida as informações necessárias, a página contém o botão "Iniciar Análise" com o propósito de inicialização da ferramenta, assim chamando a segunda página da aplicação, onde serão apresentados os gráficos com os dados coletados.

Figura 19: Página Inicial da Ferramenta

MIB	Nome	OID	Descrição
IF-MIB	ifInOctets	1.3.6.1.2.1.2.2.1.10	Taxa de bytes recebidos.
IF-MIB	ifOutOctets	1.3.6.1.2.1.2.2.1.16	Taxa de bytes enviados.
IF-MIB	ifInDiscards	1.3.6.1.2.1.2.2.1.13	Taxa de descartes de pacotes de entrada.
IF-MIB	ifOutDiscards	1.3.6.1.2.1.2.2.1.19	Taxa de descartes de pacotes de saida.
IF-MIB	ifInErrors	1.3.6.1.2.1.2.2.1.14	Taxa de pacotes com erros de entrada.
IF-MIB	ifOutErrors	1.3.6.1.2.1.2.2.1.20	Taxa de pacotes com erros de saida.
IF-MIB	ifInUnknownProtos	1.3.6.1.2.1.2.2.1.15	Taxa de pacotes de protocolos desconhecidos recebidos.
TCP-MIB	tcpAttemptFails	1.3.6.1.2.1.6.7	Número de tentativas de conexão falhadas.
TCP-MIB	tcpInErrs	1.3.6.1.2.1.6.14	Número de pacotes recebidos com erro.

Fonte: Elaborado pelo autor.

A Figura 20 mostra a segunda página da aplicação, onde contém os gráficos de monitoramento. Cada um dos gráficos representa uma ou uma dupla de OIDs mostrado na tabela. O primeiro mostra as informações da ifInOctets e ifOutOctets, o segundo mostra os ifInDiscards e ifOutDiscards, o terceiro ifInErrors e ifOutErrors, o quarto, quinto, sexto e sétimo mostram respectivamente as OIDs ifInUnknownProtos, tcpAttemptFails, tcpInErrors e tcpOutSegs. O oitavo gráfico mostra udpInDatagrams e udpOutDatagrams e o último somente o udpNoPorts.

Figura 20: Página com os Gráficos de Monitoramento

Fonte: Elaborado pelo autor.

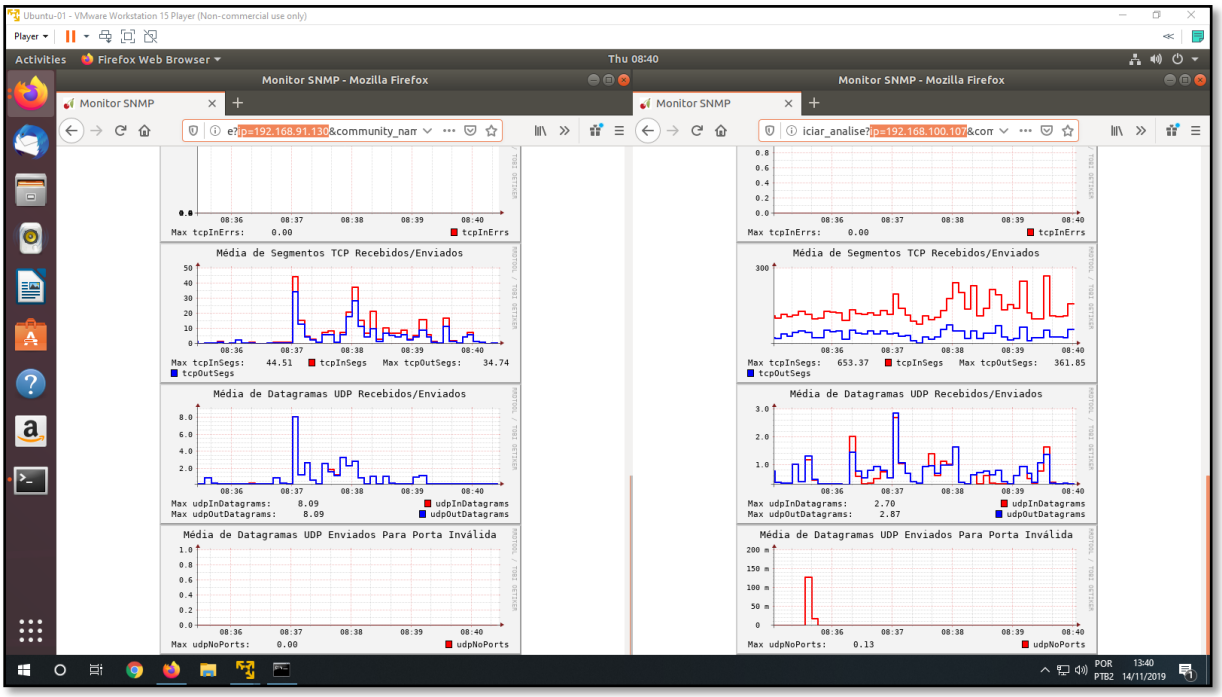
Ao acionar o botão “Parar Análise”, a coleta de dados é interrompida fazendo com que os gráficos não sejam mais atualizados, permanecendo estáticos na página. Depois de parada a coleta de dados, o usuário tem a opção de “Reiniciar a Análise” clicando no botão, fazendo com que o monitoramento reinicie do zero.

Os dados de rede estão sempre crescendo, então para fazer os gráficos o RRDtool utiliza a média dos dados coletados (por isso na Figura 20 alguns gráficos estão em branco), assim os gráficos não ficam constantemente crescendo, sendo visualmente melhor para monitorar a rede.

6.3 Casos de Teste

Um caso de utilização foi feito utilizando máquinas virtuais. Com o VMware (software de virtualização) foram criadas duas máquinas virtuais com o sistema operacional Ubuntu na versão 18.04, sendo o sistema host um Windows 10. Em um dos sistemas operacionais na máquina virtual, foi configurada a ferramenta para que ela coletasse os dados tanto do outro Ubuntu quanto do host Windows 10, já nos alvos da coleta foi configurado o protocolo SNMP para que aceitasse a requisição e mandasse os dados para o gerente. O resultado está mostrado na Figura 21, sendo os gráficos da esquerda com o IP destacado de 192.168.91.130 uma máquina Ubuntu, e o da direita com o IP de 192.168.100.107 o host Windows 10.

Figura 21: Teste utilizando máquinas virtuais

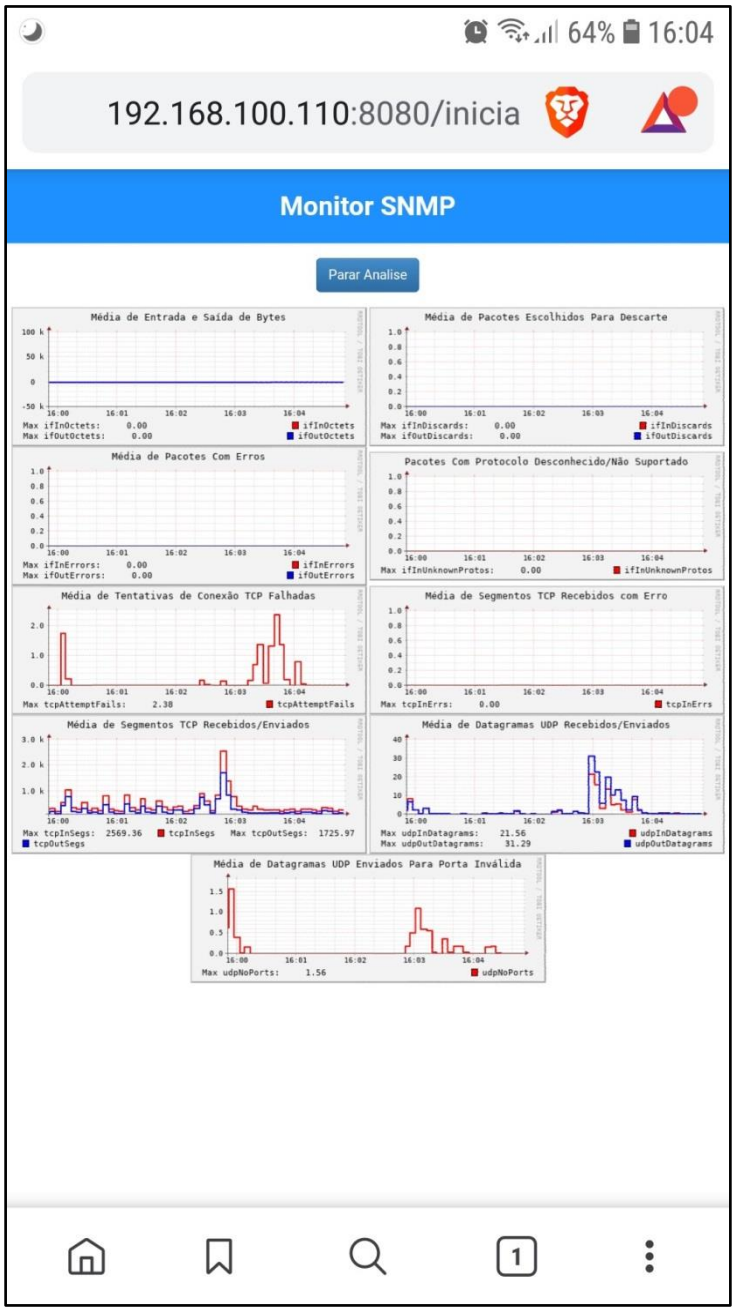


Fonte: Elaborado pelo autor.

Outro caso de teste que foi realizado utilizando uma rede doméstica e três dispositivos conectados a ela: um Desktop com sistema operacional Windows 10 com o papel de alvo do monitoramento, um Notebook com o sistema Ubuntu como servidor da ferramenta, e um Smartphone com o sistema Android acessando o site para visualização dos gráficos. A Figura 22 expõe o resultado do teste feito,

mostrando os gráficos de monitoramento do Desktop Windows 10 enquanto o mesmo estava conectado na rede.

Figura 22: Teste na rede doméstica



Fonte: Elaborado pelo autor.

Como podemos observar pelas imagens, a ferramenta gerou os gráficos de monitoramento das máquinas alvo conforme a especificação para assim elas poderem ser gerenciadas a fim de manter uma boa conexão de rede com uma taxa baixa de erros e sem lentidões na conexão.

7 Conclusão

Os principais aspectos da ferramenta desenvolvida são o monitoramento de rede e a geração de gráficos para que o administrador de redes possa avaliar e tomar medidas corretivas nessa rede de forma que ela possa permanecer segura e estável para que os usuários sejam capazes de realizar seus afazeres sem ter que enfrentar conexões de baixa qualidade.

No decorrer do cronograma de trabalho, foram estudadas todas as ferramentas necessárias para o desenvolvimento da aplicação, como conceitos de redes e protocolos, com foco no protocolo SNMP, a documentação da linguagem Python bem como a documentação das bibliotecas utilizadas. As dificuldades identificadas com o decorrer do desenvolvimento da ferramenta foram em relação à linguagem de programação, visto que o autor não possuía nenhum conhecimento prévio a respeito. Muito embora a linguagem Python realmente seja fácil de aprendizado, a biblioteca que implementa o protocolo SNMP por outro lado não é muito simples de se manipular, o que levou um tempo a mais do que o planejado o seu aprendizado.

Apesar de definido um cronograma para o projeto, as dificuldades encontradas fizeram com que o cronograma se atrasasse e fez com que alguns módulos fossem deixados para um desenvolvimento futuro. Dois desses módulos seriam a implementação de receptores de Traps que são enviadas pelo dispositivo gerenciado contendo informações sobre eventos que lhe ocorreram, e a implantação de suporte para a versão 3 do protocolo que envolve mais segurança e criptografia. Também, uma melhoria que poderia ser implementada em um trabalho futuro seria a inclusão de um número maior de OIDs para outras informações, não só relacionadas a redes, como por exemplo, o uso de CPU ou memória de um computador, ou até medição de temperatura de equipamentos específicos, deixando a escolha dessas OIDs a cargo do usuário.

A aplicação atendeu as expectativas conforme os testes realizados utilizando a rede de testes “*demo.snmplabs.com”* e uma rede doméstica como foi mostrado no capítulo de demonstração da ferramenta. Ambas funcionaram como previsto, em relação à coleta de dados e a geração/visualização dos gráficos de rede.

Referências

CHERRYPY. Documentação do CherryPy. Disponível em: <https://docs.cherrypy.org/en/latest/>. Acesso em: 19 de setembro 2019.

DIAS, Beethovem Z.; JUNIOR, Nilton A. Protocolo de Gerenciamento SNMP. Nota técnica CBPF-NT-006/01.

ETINGOF, I. Documentação do PySNMP. Disponível em: <http://snmplabs.com/pysnmp/index.html>. Acesso em: 4 de abril 2019.

GITHUB. “O que é CSS?”. Disponível em: <http://tableless.github.io/iniciantes/manual/css/>. Acesso em: 4 de abril 2019.

GUSMÃO, C. “Visão da gerência de riscos na engenharia de software”. Disponível em: <https://www.devmedia.com.br/visao-da-gerencia-de-riscos-na-engenharia-de-software/9144>. Acesso em: 13 de junho 2019.

KUROSE, Jim F; ROSS, Keith W. “Redes de Computadores e a Internet”. 6ª Edição. Pearson Education do Brasil, 2013.

LUCIDCHART. “O que é um diagrama de classe UML?”. Disponível em: <https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-classe-uml>. Acesso em: 13 de junho 2019.

LUCIDCHART. “O que é um diagrama de sequência UML?”. Disponível em: <https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-sequencia-uml>. Acesso em: 13 de junho 2019.

LUCIDCHART. “UML Use Case Diagram Tutorial”. Disponível em: <https://www.lucidchart.com/pages/uml-use-case-diagram>. Acesso em: 13 de junho 2019.

MAURO, Douglas R.; SCHMIDT, Kevin J. “Essential SNMP”. O’Reilly, 2001.

MOZILLA. “HTML: Linguagem de Marcação de Hipertexto”. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTML>. Acesso em: 4 de abril 2019.

MOZILLA. “JavaScript”. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>. Acesso em: 4 de abril 2019.

OETIKER, T. Documentação do RRDtool” Disponível em: <https://oss.oetiker.ch/rrdtool/doc/index.en.html>. Acesso em: 19 de setembro 2019.

OETIKER, T. “Multi Router Traffic Grapher”. Disponível em: <https://oss.oetiker.ch/mrtg/>. Acesso em: 4 de abril 2019.

PYTHON. Documentação da linguagem Python. Disponível em: <https://www.python.org/doc/>. Acesso em: 4 de abril 2019.

PYTHON-RRDTOOL. Documentação da biblioteca python-rrdtool. Disponível em: <https://pythonhosted.org/rrdtool/index.html>. Acesso em: 19 de setembro 2019.

SILEIKA, R. “Pro Python System Administration”. 2ª Edição. Apress, 2014.