Capítulo 1

Monitoramento de uso de memória, processador e tráfego de rede no GNU/Linux Ubuntu Server 22.04.3 LTS com Prometheus 2.45.1 e Grafana 10.1.4

Filipe Moisés Soares, Vinícius Bernardes dos Santos, Wagner Ribas Lopes de Castilho, Yasmin Devegili

Universidade do Estado de Santa Catarina - Centro de Ciências Tecnológicas (UDESC - CCT) Joinville – SC – Brasil

Resumo

O monitoramento eficiente de servidores e seus ambientes é crucial para o uso profissional da tecnologia. Diversos tipos de empreendimentos dependem de um acompanhamento rigoroso de seus servidores, visando corrigir e prevenir problemas que podem resultar em gastos operacionais e perdas financeiras. Nesse contexto,

é necessário fazer uso de ferramentas adequadas para atender essa demanda. Esse tutorial explora a aplicação da tecnologia Prometheus versão 2.45.1, devido à sua adequação e facilidade de uso. Além disso, integra a tecnologia Grafana, versão 10.1.4, a fim de aprimorar a visualização dos dados coletados, proporcionando uma compreensão mais abrangente do ambiente monitorado. A seleção criteriosa das métricas a serem monitoradas desempenha um papal muito importante nesse processo. Nesse tutorial, concentramos a atenção em métricas essenciais, como memória principal, CPU e tráfego de rede. A análise dessas métricas não apenas visa identificar problemas imediatos, mas também proporcionar informações a respeito do consumo e desempenho desses recursos na infraestrutura, contribuindo para uma gestão mais eficiente. Além disso, este tutorial é realizado em um ambiente com sistema operacional GNU/Linux Ubuntu Server 22.04.3 LTS, essa escolha é estratégica considerando a estabilidade e a confiabilidade desse sistema operacional voltado para ambientes de servidores. O objetivo deste tutorial é exemplificar o monitoramento de consumo de memória, processador e tráfego de rede de um sistema operacional GNU/Linux Ubuntu Server 22.04.3 LTS, por meio de um caso de uso com Prometheus 2.45.1 e Grafana 10.1.4 (Edição Enterprise). Além da descrição do cenário, são apresentados os resultados experimentais obtidos em um ambiente local.

1. Monitoramento de recursos em servidores web

O monitoramento do servidor é parte essencial da arquitetura de qualquer aplicação. O monitoramento consiste em coletar dados do sistema operacional, e as métricas do seu hardware, de forma constante e em tempo real. Ao coletar esses dados é possível gerar estatísticas em tempo real e analisar o desempenho da rede. Com isso, qualquer interrupção ou anomalia que surgir pode ser tratada pelo responsável antes que cause maiores danos. Isso garante a estabilidade e a confiabilidade das operações do servidor, contribuindo para uma experiência mais eficiente e segura para os usuários da aplicação (RENITA; ELIZABETH, 2017).

A capacidade do servidor depende diretamente dos seus dispositivos físicos. Esses dispositivos são conhecidos como hardware e definem restrições como capacidade de processamento, armazenamento e entrada/saída de dados. Assim, cada componente tem sua função, e quando integrados podem executar as tarefas que o sistema operacional necessita. Os principais componentes são a unidade central de processamento (CPU), a memória principal e dispositivos de entrada e saída (CARVALHO; LORENA, 2016).

Em servidores *web*, o monitoramento dos recursos de CPU, memória e tráfego de rede são importantes pilares para garantir a disponibilidade e o desempenho do servidor e são fundamentais o seu funcionamento. A observação destes recursos em tempo real permite

identificar e resolver problemas antes que eles causem interrupções ou quedas no desempenho. Os principais servidores *web* são Nginx, Microsoft IIS, LiteSpeed Web Server e Apache, esse último é o mais popular do mundo, responsável por hospedar mais de 40% dos sites da internet (THE APACHE SOFTWARE FOUNDATION, 2023). É um servidor gratuito e de código aberto, que oferece uma gama de recursos e funcionalidades. Por esses motivos, neste tutorial é utilizado o servidor Apache 2.4.52 para realizar o estudo de caso.

1.1. Métricas

Para garantir a eficiência do monitoramento em servidores, é crucial estabelecer identificar as métricas que melhor contribuem para a avaliação do desempenho do sistema. Portanto, faz-se necessário empregar ferramentas de monitoramento que possam capturar métricas relevantes provenientes do sistema operacional ou hardware do servidor.

O desempenho de um servidor está intrinsecamente ligado à sua capacidade de utilizar recursos, sendo assim, a análise das métricas concentra-se principalmente na utilização de recursos como CPU, interface de rede e discos (OHTA, 2016).

Ao analisar as tarefas e programas executados pelo servidor, é essencial monitorar a utilização da memória principal, o desempenho de CPU nas operações e as variações, bem como a confiabilidade do tráfego de rede. Essa abordagem abrangente das métricas do sistema

resulta na administração proativa e otimização contínua do ambiente do servidor.

Memória principal: A memória principal desempenha um papel crucial no monitoramento de servidores, sendo considerada uma das métricas mais significativas. Apesar de possuir maior capacidade de armazenamento, quando comparada com a memória *cache*, a alocação eficiente de memória principal é vital para garantir o funcionamento ininterrupto dos processos. Quando um processo atinge o limite de espaço disponível na memória, ocorre a interrupção, resultando na finalização forçada desse processo. Essa situação de processos encerrados de forma abrupta não apenas representa uma perda de dados, mas também acarreta prejuízos em termos de tempo e energia investidos no desenvolvimento e execução desses processos (RUAN, et al. 2020). Nesse contexto, compreender e monitorar de maneira abrangente a utilização da memória principal é fundamental para otimizar o desempenho do servidor e prevenir possíveis interrupções indesejadas nos processos em execução.

Em servidores *web*, é comum desenvolvedores iniciantes deixarem algumas brechas de segurança em seu código. O erro mais comum cometido em programação é o vazamento de memória, que ocorre quando um programa não libera a memória que alocou, mesmo quando ela não é mais necessária. (TOMMCDON, 2023). Se um servidor *web* tem monitoramento do recurso de memória, é fácil identificar quando isto está acontecendo pois o gráfico que indica o consumo de memória mostra

um comportamento que é comum neste tipo de problema (JETBRAINS, 2023).

Como é possível observar na Figura 1.1, quando ocorre um vazamento de memória, o indicador do gráfico sobe até o nível crítico. É possível observar que quase não há diminuição no consumo total, pois a memória não está sendo liberada, até chegar no limite do servidor e este ser reiniciado.

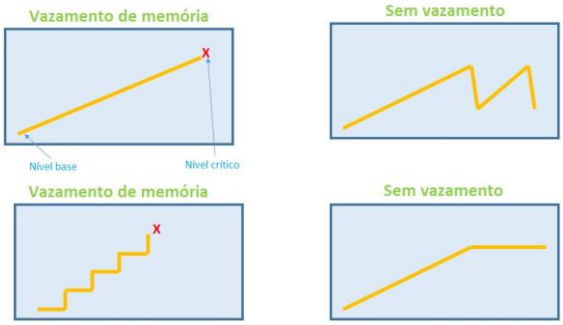

Figura 1.1 - Gráficos do vazamento de memória no servidor.

Fonte: (adaptado de ALIOSTAD, 2023)

CPU: A unidade central de processamento *(CPU)* representa um componente vital para o desempenho eficaz de servidores. Durante sua operação, a CPU assume a responsabilidade de realizar o ciclo de busca-decodificação-execução, recuperando instruções armazenadas na memória principal do servidor para executar operações sobre os dados. A velocidade com que

a CPU processa informações e a quantidade de operações realizadas por segundo são fatores determinantes para a agilidade do servidor (CARVALHO; LORENA, 2016).

Ao incorporar essa métrica ao contexto de monitoramento, é possível avaliar de maneira contínua e proativa o comportamento da CPU em tempo real. Monitorar a utilização da CPU fornece informações valiosas sobre a carga de trabalho do servidor, identificando possíveis gargalos de desempenho e otimizando a distribuição de recursos computacionais. Essa abordagem permite uma administração mais eficiente, antecipando-se a eventuais quedas de desempenho e garantindo uma resposta ágil às demandas operacionais.

Tráfego de rede: No âmbito do monitoramento abrangente, o tráfego de rede emerge como um fator de grande impacto nos servidores *web*. Devido à sua intensa demanda, suscetibilidade a ameaças e riscos de ataques à rede, torna-se necessário implementar um monitoramento robusto, apoiado por ferramentas especializadas. No cenário atual, o aumento da velocidade das redes sem fio acentua a necessidade de estabilidade, demandando a identificação e correção proativas de potenciais problemas de rede para prevenir falhas potencialmente desastrosas mesmo em pequenos períodos de tempo (SO-IN, 2006).

Os agentes de monitoramento desempenham um papel vital nesse contexto, exigindo capacidades para

identificar, isolar e corrigir irregularidades no tráfego de rede. Essa abordagem estratégica visa aprimorar a integridade operacional dos servidores *web*, garantindo a continuidade e eficiência das operações.

1.2. GNU/Linux Ubuntu Server

O Ubuntu Server, lançado pela Canonical Ltd. em outubro de 2004, destaca-se como uma das distribuições Linux mais adotadas. Sob a direção de Mark Shuttleworth, este sistema operacional de código aberto ganhou reconhecimento mundial por sua robusta comunidade de suporte e ênfase na facilidade de uso. Ao decorrer do tempo, o Ubuntu Server consolidou sua posição como uma escolha confiável para uma variedade diversificada de implementações de servidores (CANONICAL, 2023).

As principais diferenças entre as versões *Desktop* e *Server*, em essência, estão relacionadas às aplicações pré-instaladas, sendo mínimas na versão *Server*; e interface gráfica *(GUI - Graphical User Interface)*, com a versão *Server* focando exclusivamente na utilização da interface por linha de comando (*CLI - Command-Line Interface)* (CAWLEY, 2023).

Para o desenvolvimento do trabalho, foi utilizado a versão GNU/Linux Ubuntu Server 22.04.3 LTS, a mais recente atualmente, com manutenção de segurança prevista até 2032. Dessa forma, garante segurança, manutenção e suporte para o desenvolvimento. Ademais, é relevante a utilização da versão *Server* a fim de priorizar

o desempenho e a confiabilidade no contexto de testes, análises e resultados esperados ao decorrer do trabalho.

2. Ferramentas de monitoramento

Ao selecionar a ferramenta mais apropriada para o monitoramento de sistemas, vários fatores devem ser considerados, incluindo funcionalidades, custos e a relevância das métricas a serem analisadas. A análise cuidadosa desses elementos é crucial para a formulação de estratégias eficazes e para assegurar que o desempenho do hardware atenda às expectativas.

No contexto do monitoramento, várias ferramentas se destacam, cada uma oferecendo abordagens distintas. Por exemplo, o Datadog é conhecido por sua versatilidade, oferecendo uma conta gratuita para até cinco hosts e se destacando no monitoramento de infraestrutura, plataformas, aplicações e serviços na nuvem. Outra opção notável é o Zabbix, que se destaca por sua flexibilidade no monitoramento de infraestrutura, abrangendo servidores, máquinas virtuais e redes (FOSTER, 2022).

Para este tutorial, optamos por focar na ferramenta Prometheus, conhecida por manter registros de métricas em tempo real e pela capacidade ágil de diagnóstico. Além disso, o Prometheus pode ser integrado com o Grafana, que oferece gráficos interativos e painéis personalizáveis. Essa combinação proporciona uma solução abrangente para o monitoramento eficaz de sistemas.

2.1. Prometheus

O Prometheus é uma ferramenta *open-source* adotada no campo da tecnologia da informação, destacando-se pelo seu uso em empresas notáveis, como Docker, JustWatch e GrafanaLabs. Originalmente desenvolvido pelo SoundCloud em 2012, o Prometheus é atualmente supervisionado pela Cloud Native Computing Foundation, a mesma entidade responsável pelo Kubernetes. Sua versão mais recente é a 2.47.2, com a versão 2.48.0 em pré-lançamento (PROMETHEUS, 2023).

Uma das principais características do Prometheus é a sua capacidade de aprimorar o monitoramento de sistemas de forma eficiente e versátil. Isso se deve em grande parte à criação de sua própria linguagem de consulta, o PromQL, que possibilita análises, consultas e operações simplificadas. Além disso, o Prometheus utiliza um modelo de dados que facilita a criação de alertas, notificações, gráficos e tabelas com base nas métricas monitoradas (BRAZIL, 2018). Essas características fazem do Prometheus uma escolha popular entre profissionais da área, principalmente devido à sua integração com outras aplicações, incluindo o Grafana (PROMETHEUS, 2023).

Classificado como *Whitebox Monitoring* o Prometheus opera com informações provenientes do sistema operacional, como CPU, memória e outras métricas abordadas nesse tutorial. Em contrapartida,

existe também a *Blackbox Monitoring*, definido como monitoramento do sistema a partir do exterior, sem possuir conhecimento interno detalhado. Esse tipo de monitoramento externo é realizado, por exemplo, por meio de requisições HTTP e *pings*, sem uma visão direta sobre o funcionamento interno do sistema (BEHARA, 2019).

O núcleo do ecossistema do Prometheus é o servidor Prometheus, responsável por coletar e armazenar métricas do ambiente. Esse servidor é configurável, o que permite adaptações para atender a diversas necessidades. A coleta de métricas ocorre de maneira contínua, fornecendo informações em tempo real sobre o estado do sistema (PROMETHEUS, 2023).

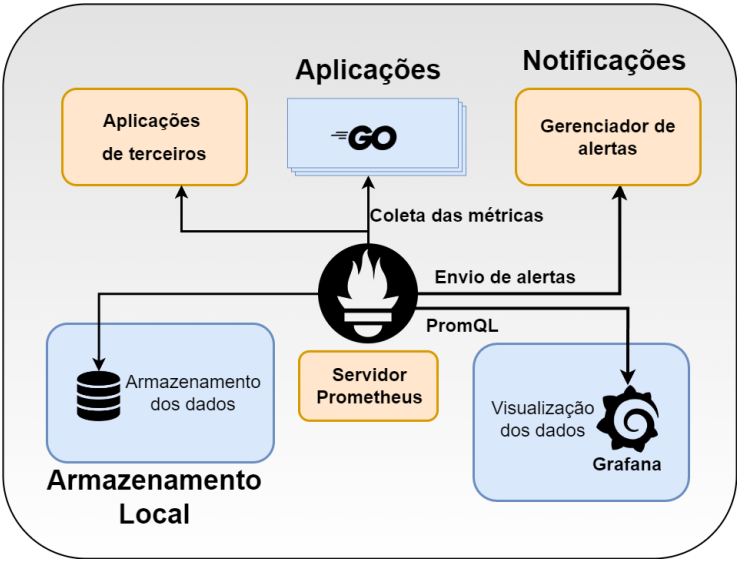

Figura 2.1 - Arquitetura do Prometheus.

Fonte: (próprios Autores)

Na Figura 2.1, é possível observar os passos envolvidos no processo de monitoramento com o Prometheus. As aplicações monitoradas pelo Prometheus fazem uso de *Clients Libraries*, que adicionam as instruções necessárias ao código. Essas bibliotecas são escolhidas de acordo com a linguagem de programação utilizada, sendo oficialmente mantidas pelo Prometheus: Go (utilizada no desenvolvimento da linguagem de consultas PromQL), Java, Scala, Python, Ruby e Rust. Além dessas, é possível importar muitas outras bibliotecas de terceiros (não oficiais) (PROMETHEUS, 2023).

Em situações em que o acesso direto às aplicações não é possível, são utilizados *Exporters* em vez de *Client Libraries.* Esses *Exporters* capturam as métricas solicitadas das aplicações por meio de requisições do servidor Prometheus no formato desejado, conforme elucidado na Figura 2.1 pelo bloco "Aplicações de terceiros" (IBM, 2023). Em relação ao armazenamento local, os dados são alocados de maneira personalizada, recebendo diversas amostras por segundo, mantendo a simplicidade e confiabilidade. Além disso, é possível configurar o armazenamento para longos períodos (FRISVOLD, 2020).

A escolha do Prometheus foi feita levando em conta os seguintes critérios: ser compatível com o GNU/Linux, ter licença *open-source,* ser gratuito e ter uma conectividade com ambiente externo via HTTP,

sendo facilmente integrado ao Grafana. Acrescenta-se o fato de fazer parte dos projetos da CNCF (*Cloud Native Computing Foundation)*, o que reforça a escolha do Prometheus para ambientes que demandam confiabilidade e desempenho em servidores *web* e sistemas de monitoramento avançado.

2.2. Grafana

O Grafana, desenvolvido pela Grafana Labs e lançado por Torkel Ödegaard em 2013, tem como objetivo tornar o acesso, o gerenciamento de métricas e a criação de *dashboards* interativos acessíveis a um público amplo (DAM, 2019). Com sua abordagem simples e uma interface visual amigável, o Grafana se destaca como uma ferramenta de escolha para a visualização de dados e o monitoramento de sistemas (ÖDEGAARD, 2019).

O Grafana possibilita a exploração de métricas, a criação de alertas e a visualização de *logs*. Sua funcionalidade mais reconhecida é a capacidade de criar *dashboards* personalizáveis que agregam dados de diversas fontes, transformando a maneira como é visualizado os dados coletados de aplicações (GRAFANA LABS, 2023).

Figura 2.2 - *Dashboard* no Grafana.

Fonte: (COCK, 2023)

Conforme evidenciado na Figura 2.2, a integração entre Prometheus e Grafana oferece a capacidade de criar painéis abrangentes e informativos. Esses painéis podem conter diversos tipos de visualizações, incluindo gráficos de linhas, gráficos de barras, tabelas e outros elementos, proporcionando uma ampla gama de opções de representação visual dos dados. Além disso, a integração oferece uma série de recursos de filtragem, permitindo a otimização da visualização para atender às necessidades específicas de monitoramento e análise (IBM, 2023). Dessa forma, os usuários podem personalizar seus painéis de acordo com as métricas e informações que são mais relevantes para o seu ambiente, proporcionando uma experiência de monitoramento adaptável e informativa.

O Grafana tem grande capacidade de integração com diversas plataformas. Ele não se restringe à coleta de métricas em ambientes tradicionais, como o Kubernetes, mas permite uma integração facilitada com outras tecnologias, como Raspberry Pi e até mesmo o Google Sheets (GRAFANA LABS, 2023).

A escolha do Grafana como ferramenta de visualização foi baseada nos seguintes motivos: ser compatível com o GNU/Linux, ter licença *open-source*, ser gratuito, possuir interface *web*, ser customizável e ter facilidade de integração com diversas origens de dados, inclusive Prometheus.

2.3. Cenário

Para exemplificar o monitoramento de um servidor em condições próximas à um cenário real, é utilizada a arquitetura mostrada na Figura 2.3. Com a utilização de uma máquina virtual (MV) KVM 6.2.0 Tipo 2 para instalar o sistema operacional (SO) hóspede GNU/Linux Ubuntu Server 22.04.3 LTS, as métricas monitoradas desse ambiente não sofrerão interferências do sistema hospedeiro, isso porque a MV tem seu próprio hardware virtualizado (KVM, 2023).

Dois componentes são importantes nesse sistema: o servidor Apache 2.4.52 e o Prometheus Node Exporter 1.6.1. O primeiro é importante pois utiliza os recursos de hardware da MV monitorada, o segundo tem o objetivo de capturar as métricas e fornecer via HTTP para o servidor Prometheus 2.45.1.

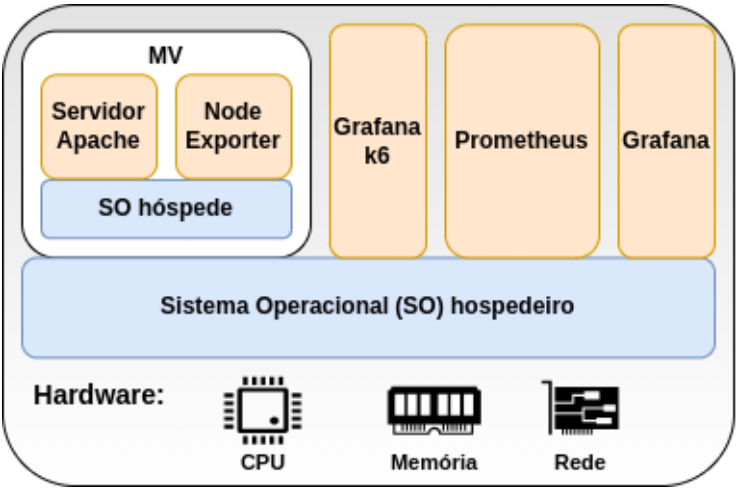

Figura 2.3 - Arquitetura do cenário.

Fonte: (próprios Autores)

No SO hospedeiro, GNU/Linux Ubuntu Desktop 22.04.3 LTS, são executadas três ferramentas: Prometheus 2.45.1, Grafana 10.1.4 (Edição Enterprise) e o Grafana k6 0.46.0. Como explicado nessa seção, o Prometheus e o Grafana têm o papel de manipular as métricas e expô-las graficamente, já o Grafana k6 tem o papel de executar o plano de testes, simulando as requisições dos usuários ao servidor.

O plano testes realizado é o teste de carga média. Testes de carga média têm o papel de simular o tráfego em um cenário de uso comum da aplicação. Esse tipo de teste é divido em de três períodos: aumentar o número de usuários simultâneos gradualmente, com duração de até quinze porcento do tempo de teste total, até atingir o pico

usuários desejado; manter esse estado por um período maior que o anterior; e por fim, diminuir gradualmente essa quantidade para zero, durante o mesmo tempo do primeiro período (GRAFANA LABS, 2023).

Após a realização de alguns testes, o plano de testes foi definido para se adequar contexto desse tutorial e está ilustrado na Figura 2.4. São usados cento e cinquenta usuários simultâneos e tempo total de teste de oitenta segundos.

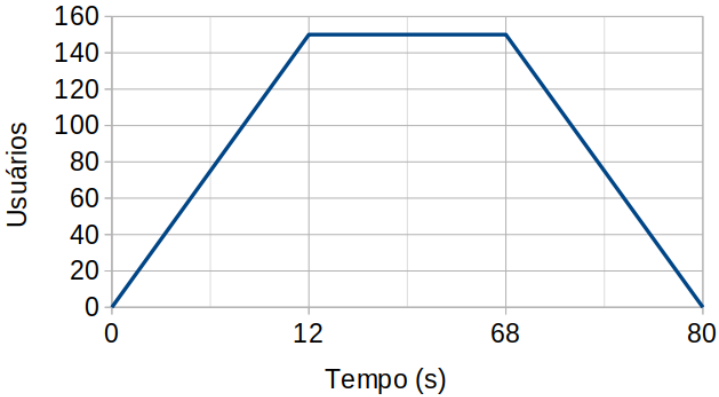

Figura 2.4 - Plano de testes.

Fonte: (próprios Autores)

Com esse cenário e plano de testes, é possível visualizar as mudanças nos gráficos do Grafana, que representam o uso dos recursos necessários para suprir a demanda de requisições durante o teste. Como já mencionado na Seção 1, em um ambiente de servidor, os três recursos essenciais no monitoramento de um servidor

considerados nesse tutorial são: o uso da CPU, memória principal e tráfego de rede.

3. Ambiente de testes

Para a instalação e configuração do ambiente que são executados os testes é preciso seguir o fluxo que está ilustrado na Figura 3.1. Para a criação da MV, são utilizadas duas tecnologias: KVM e QEMU, que trabalham em conjunto para virtualizar ambientes hóspedes. Além disso, o Virt-manager 4.0.0 é utilizado para gerenciar esses ambientes através de uma interface gráfica (CANONICAL, 2023).

Desse modo, o primeiro passo é instalar as bibliotecas necessárias para a criação da MV, seguido da instalação do Virt-manager e instalação do sistema GNU/Linux Ubuntu Server 22.04.3 LTS. Nesse sistema, é instalado o Prometheus Node Exporter 1.6.1 e o servidor Apache 2.4.52, seguido da configuração do arquivo servido.

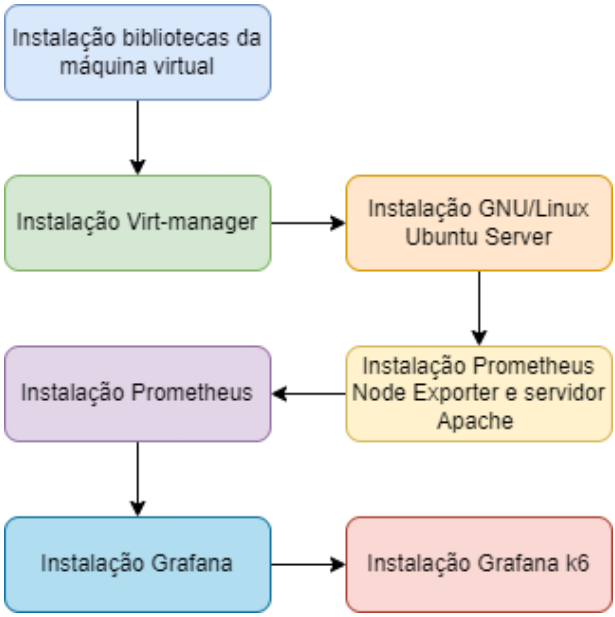

Figura 3.1 - Fluxo de configuração do ambiente.

Fonte: (próprios Autores)

Após a finalização da instalação e configuração do sistema monitorado, é possível instalar o Prometheus 2.45.1 no SO hospedeiro e configurá-lo utilizando o endereço *IP* da MV e a porta exposta pelo Node Exporter. Por fim, sem uma ordem específica, são instalados o Grafana 10.1.4 (Edição *Enterprise*) e o Grafana k6 0.46.0. O fluxo foi definido levando em consideração as dependências entre as aplicações e a otimização do tempo, evitando trocas de SO.

3.1. Instalação da Máquina Virtual e Ubuntu Server

Para iniciar, os comandos do Código-Fonte 3.1 devem ser executados para instalar os pacotes do que vão possibilitar a criação da máquina virtual (CANONICAL, 2023).

Código-Fonte 3.1 Instalação do KVM

```
$ sudo apt update

$ sudo apt install qemu-kvm libvirt-
daemon-system
```

Após a instalação da KVM, deve-se seguir os comandos do Código-Fonte 3.2 para instalar o Virt-Manager.

Código-Fonte 3.2 Instalação do Virt-Manager

```
$ sudo apt install virt-manager
```

Com isso, o sistema operacional GNU/Linux Ubuntu Server 22.04.3 LTS já pode ser instalado na MV. O primeiro passo é obter a imagem do sistema através do site oficial do Ubuntu (CANONICAL, 2023). Feito isso, o próximo passo é abrir o Virt-Manager e clicar no primeiro botão à esquerda mostrado na Figura 3.2, para criar uma máquina virtual.

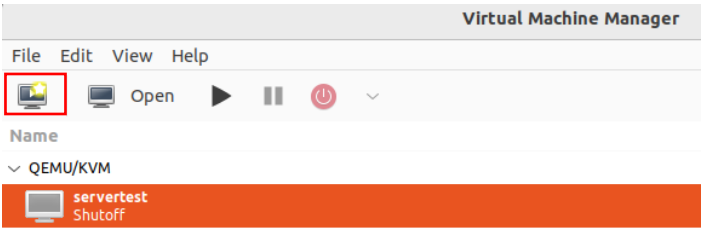

Figura 3.2 - Criação da máquina virtual.

Fonte: (próprios Autores)

Uma janela irá aparecer e a primeira opção deve ser selecionada para a instalação ser feita através imagem do Ubuntu obtida anteriormente, conforme a Figura 3.3.

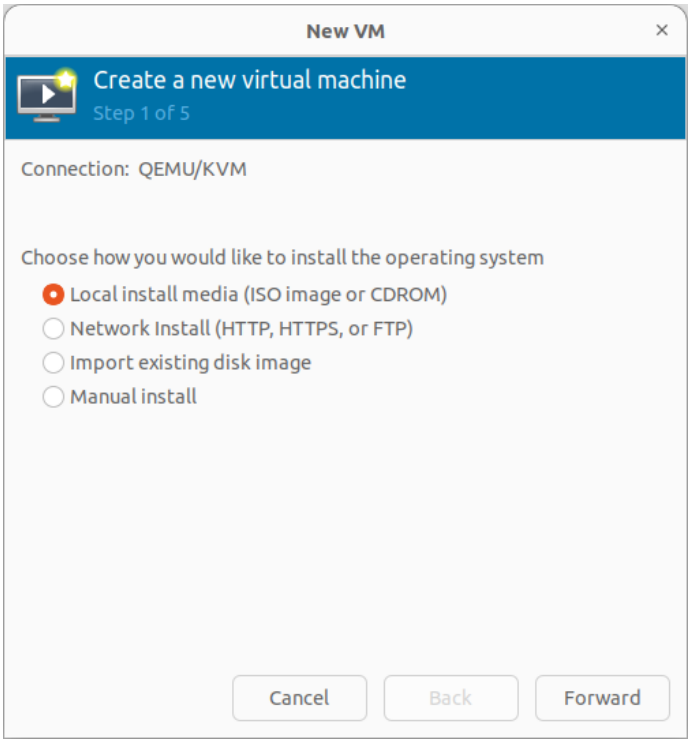

Figura 3.3 - Instalação por meio da imagem.

Fonte: (próprios Autores)

Na próxima janela deve-se clicar em "*Browser"*, localizar a imagem obtida anteriormente e clicar na opção *"Choose Volume"*. Com a imagem selecionada, e preciso proceder para a instalação clicando em "*Forward"*. No próximo passo, é configurado a quantidade de 2048 MB para memória principal e uma CPU destinada ao sistema, conforme mostrado na Figura 3.4.

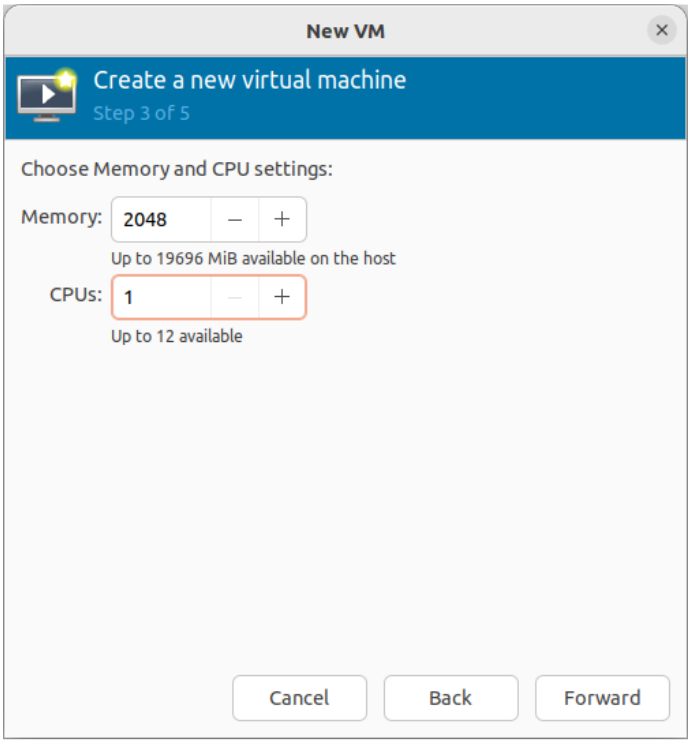

Figura 3.4 – Dimensionamento da memória e processamento.

Fonte: (próprios Autores)

Em seguida, deve-se manter a primeira opção selecionada e informe a quantidade de armazenamento que a máquina virtual utiliza, que nesse caso é de 10 GiB (*Gibibyte*), conforme a Figura 3.5.

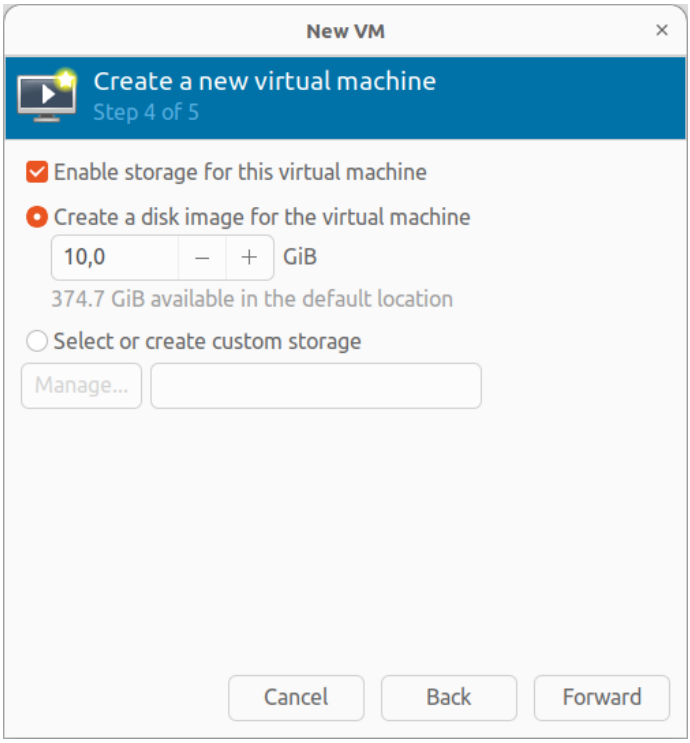

Figura 3.5 – Dimensionamento do armazenamento.
Fonte: (próprios Autores)

Na próxima janela, é mostrado resumo das configurações escolhidas, confira se estão corretas e finalize a instalação clicando em "*Finish*". Na sequência, uma nova janela abrirá, com o gerenciador de inicialização, seleciona-se a primeira opção para proceder com a instalação do sistema. As configurações de hardware foram escolhidas de acordo com os requisitos mínimos do GNU/Linux Ubuntu Server. Além disso, é

seguido o passo a passo da instalação básica fornecido pelo site oficial do Ubuntu (CANONICAL, 2023), tendo um único passo adicional que é habilitar a instalação do servidor OpenSSH, no passo "*SSH Setup*".

3.2. Instalação do servidor Apache e Prometheus Node Exporter

Com o SO hóspede instalado, é possível fazer a instalação do Prometheus Node Exporter 1.6.1. O Código-Fonte 3.3 lista os passos de como obter e extrair o programa para ser executado posteriormente (PROMETHEUS, 2023).

Código-Fonte 3.3 Instalação do Prometheus Node Exporter

```
$ wget
https://github.com/prometheus/node_exporter
/releases/download/v1.6.1/node_exporter-
1.6.1.linux-amd64.tar.gz

$ tar xvfz node_exporter-1.6.1.linux-
amd64.tar.gz
```

Para instalar o servidor Apache 2.4.52 e o PHP 8.1.2, são utilizados os comandos mostrados no Código-Fonte 3.4. Primeiramente, é instalado o servidor, seguido da instalação das bibliotecas do PHP (CANONICAL, 2023).

Código-Fonte 3.4 Instalação do servidor Apache e PHP

```
$ sudo apt update
$ sudo apt install apache2
$ sudo apt install php libapache2-mod-php
```

Para a aplicação funcionar corretamente, é preciso instalar o pacote GMP do PHP, além de adicioná-lo no arquivo de configuração do PHP. No Código-Fonte 3.5, após instalar o pacote, deve-se executar o segundo comando para editar o arquivo e descomente a linha que contém o código “extension=gmp” (THE PHP GROUP, 2023). Por fim, o Apache deve ser reiniciado para que consiga executar os scripts da aplicação.

Código-Fonte 3.5 Configurando o PHP

```
$ sudo apt-get install php-gmp
$ sudo nano /etc/php/8.1/cli/php.ini
$ sudo systemctl restart apache2.service
```

Com o PHP configurado, é possível acessar o diretório que ficam os arquivos servidos, criar um novo arquivo com o código disponível no repositório[1] e remover o arquivo criado na instalação do Apache. Os comandos para executar esses passos são descritos no Código-Fonte 3.6.

Código-Fonte 3.6 Configuração da aplicação

[1] https://github.com/viniciusbe/ubuntu-monitoring

```
$ cd /var/www/html

$ sudo touch index.php

$ sudo nano index.php
```

Completado esse último passo, está finalizada a configuração do SO hóspede, restando a instalação das ferramentas no SO hospedeiro.

3.3. Instalação do Prometheus, Grafana e Grafana k6

Para configurar o Prometheus no SO hospedeiro, é seguido os comandos descritos no Código-Fonte 3.7 para obter e extrair. Além disso, é preciso alterar o arquivo de configuração para informar ao Prometheus o local e como coletar as métricas (PROMETHEUS, 2023). Esse arquivo é o "prometheus.yml" e está disponível no repositório[2], necessitando alterar apenas o endereço de *IP* da respectiva MV.

Código-Fonte 3.7 Instalação do Prometheus

```
$ wget
https://github.com/prometheus/prometheus/re
leases/download/v2.45.1/prometheus-
2.45.1.linux-amd64.tar.gz

$ tar xvf prometheus-2.45.1.linux-
amd64.tar.gz
```

[2] https://github.com/viniciusbe/ubuntu-monitoring

Em seguida, deve ser usado os comandos do Código-Fonte 3.8 para obter e instalar o Grafana 10.1.4 (Edição *Enterprise*) (GRAFANA LABS, 2023).

Código-Fonte 3.8 Instalação do Grafana

```
$ sudo apt-get install -y adduser
libfontconfig1 musl

$ wget
https://dl.grafana.com/enterprise/release/g
rafana-enterprise_10.1.4_amd64.deb

$ sudo dpkg -i grafana-
enterprise_10.1.4_amd64.deb
```

Por fim, para a instalação do Grafana k6 0.46.0, devem ser executados os comandos do Código-Fonte 3.9. Para configurar e executar o teste pode ser criado um arquivo JavaScript (GRAFANA LABS, 2023). O arquivo usado neste tutorial está disponível no repositório[3], devendo-se alterar apenas o endereço de *IP* da MV.

Código-Fonte 3.9 Instalação do Grafana k6

[3] https://github.com/viniciusbe/ubuntu-monitoring

```
$ sudo gpg -k

$ sudo gpg --no-default-keyring --keyring
/usr/share/keyrings/k6-archive-keyring.gpg
--keyserver hkp://keyserver.ubuntu.com:80 -
-recv-keys
C5AD17C747E3415A3642D57D77C6C491D6AC1D69

$ echo "deb [signed-
by=/usr/share/keyrings/k6-archive-
keyring.gpg] https://dl.k6.io/deb stable
main" | sudo tee
/etc/apt/sources.list.d/k6.list

$ sudo apt-get update

$ sudo apt-get install k6
```

Com o término da instalação do Grafana k6, todo o ambiente está configurado e pronto para a realização dos testes. Portanto, as métricas do servidor podem ser coletadas e visualizadas.

4. Estudo de caso com Prometheus e Grafana

Para realizar o estudo de caso corretamente, é importante seguir um fluxo de execução das ferramentas. Primeiramente, é preciso executar o arquivo *"node_exporter",* localizado dentro diretório gerado na extração do Código-Fonte 3.3. Enquanto estiver executando, o Node Exporter servirá as métricas pelo protocolo HTTP, por padrão na porta 9100, para assim o servidor do Prometheus coletar as métricas (PROMETHEUS, 2023). Portanto, deve ser executado o

comando do Código-Fonte 3.8 para inicializar o servidor Prometheus usando o arquivo de configuração já criado na Seção 3.2.

Código-Fonte 4.1 Execução do Prometheus

```
$ ./prometheus --
config.file=./prometheus.yml
```

O próximo passo é executar o Grafana, a partir dos comandos do Código-Fonte 3.9. Feito isso, a aplicação pode ser acessada no navegador pela porta 3000, o usuário padrão é o "admin", com senha também "admin" (GRAFANA LABS, 2023). Através dessa interface gráfica, é possível criar a conexão com o Prometheus e criar o *dashboard* para visualizar os gráficos.

Código-Fonte 4.2 Execução do Grafana

```
$ sudo systemctl daemon-reload

$ sudo systemctl start grafana-server

$ sudo systemctl status grafana-server
```

Para conectar o Grafana ao Prometheus é preciso configurá-lo como uma fonte da dados no Grafana. Isso pode feito através do menu principal no canto superior esquerdo, subitem "*Data Sources*" dentro do item "*Connections*", conforme a Figura 4.1.

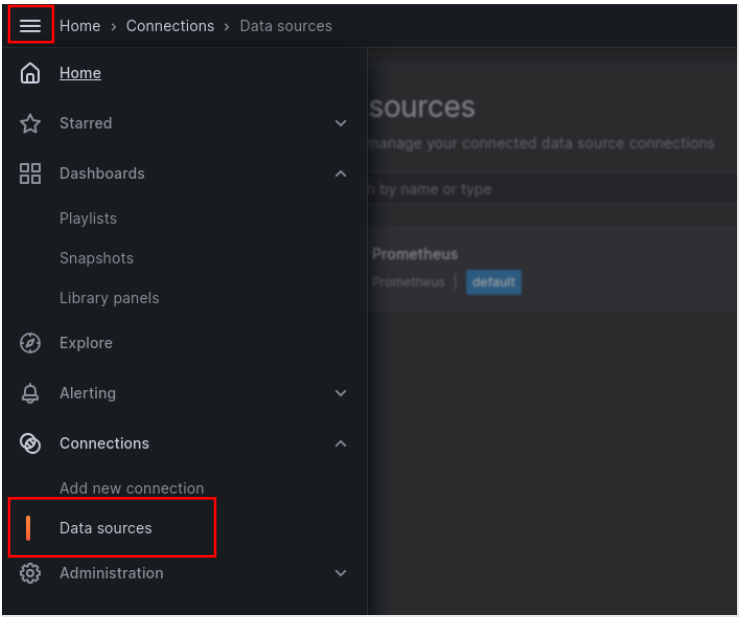

Figura 4.1 – Opção *Data Sources* no menu principal.

Fonte: (próprios Autores)

É preciso clicar no botão "*Add data source*", selecionar o tipo "Prometheus", inserir a *URL*[4] do servidor Prometheus no campo "Prometheus *server URL*" e salvar as configurações no final da página no botão "*Save & test*" (PROMETHEUS, 2023). Uma mensagem de sucesso deve aparecer se a conexão estiver funcionando corretamente, conforme a Figura 4.2.

[4] http://localhost:9090/

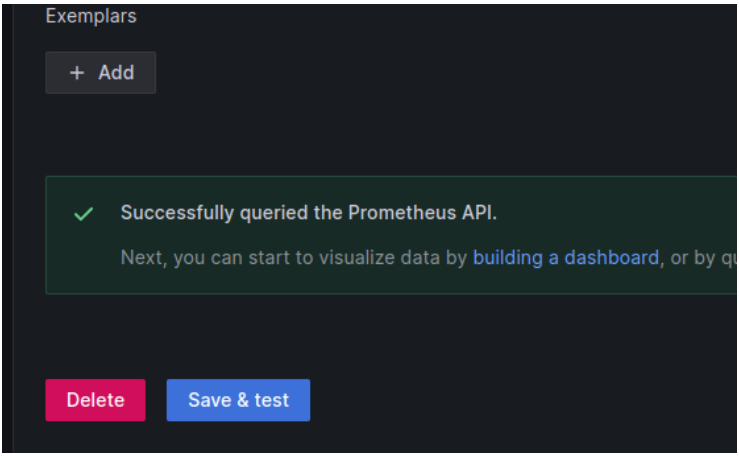

Figura 4.2 – Mensagem de sucesso.

Fonte: (próprios Autores)

Com a fonte de dados configurada, o próximo passo é importar o *dashboard* fornecido pelo Grafana com as métricas fornecidas do Node Exporter. Esse *dashboard* já é configurado para exibir as métricas mais comuns no contexto no Node Exporter, inclusive as métricas monitoradas neste tutorial (GRAFANA LABS, 2023). Para realizar esse passo, é preciso abrir novamente o menu principal, clicar no item "*Dashboards*". Em seguida, dentro da página de "*Dashboards"*, clicar na opção *"New"* e *"Import",* como ilustrado na Figura 4.3.

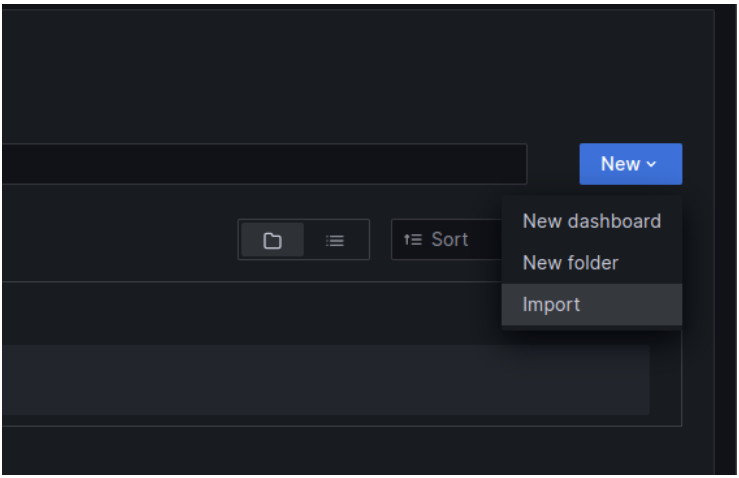

Figura 4.3 – Opção importar *Dashboard*.

Fonte: (próprios Autores)

Na próxima página, inserir o valor 1860 no campo "*Import* via grafana.com" e clicar em "*Load*". Na próxima página, ilustrada na Figura 4.4, selecione a fonte de dados "Prometheus", criada no passo anterior e finalize a importação clicando em "*Import*".

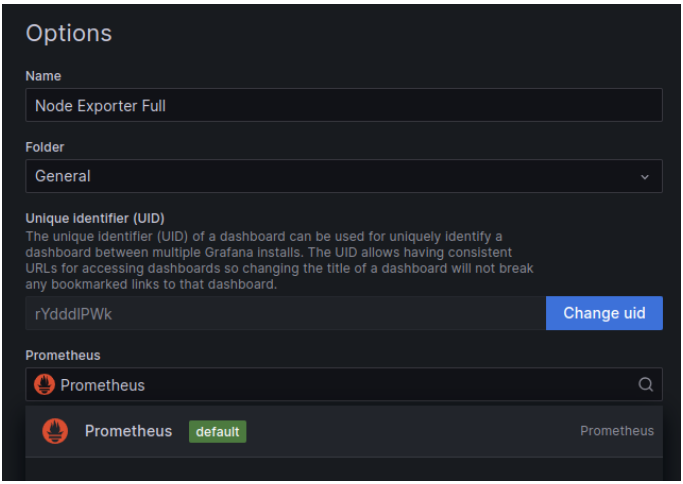

Figura 4.4 – Opções para importação do *Dashboard*.

Fonte: (próprios Autores)

Após finalizar a importação, a página do *dashboard* deve aparecer, contendo os gráficos mostrados na Figura 4.5. Um sinal que o ambiente está operando corretamente é valor do indicador "*RAM Used*", que mostra a quantidade de memória principal usada. Este valor deve ser maior que zero, considerando que o SO consome certa quantidade de memória da MV, assim como o servidor Apache.

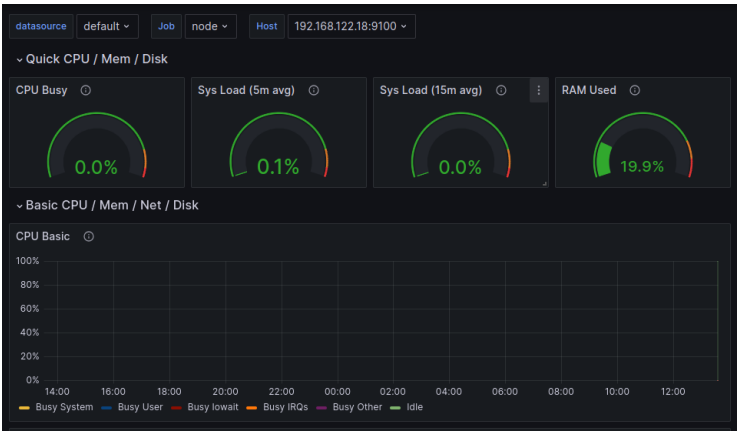

Figura 4.5 – *Dashboard* no Grafana.

Fonte: (próprios Autores)

Para executar o teste de carga média, deve-se executar o comando do Código-Fonte 4.3, que executa o arquivo de testes criado na Seção 3.2. Durante e após finalizar o teste, uma mensagem é mostrada no terminal de comandos (GRAFANA LABS, 2023). É importante notar se a mensagem na penúltima linha contém "*0 interruted interations*", indicando que o teste foi realizado com sucesso.

Código-Fonte 4.3 Execução do Grafana

```
$ k6 run <nome_do_arquivo>.js
```

Durante a execução do teste, é possível verificar as alterações nos gráficos do Grafana. Com a análise desses gráficos, é possível ter uma clara visão de como os recursos de hardware estão sendo utilizados. Desse modo,

é possível identificar e prevenir falhas ou interrupções no sistema. Mais detalhes sobre os gráficos obtidos são explicados na próxima seção.

4.1. Resultados

Nos gráficos mostrados na Figuras 4.6, 4.7 e 4.8 é possível visualizar as alterações das métricas monitoradas durante a realização do teste. O teste inicia-se no horário 18:28:15 e tem a duração total de oitenta segundos, como explicado na Seção 2.3. A Figura 4.6 refere-se ao uso de CPU da MV, sendo possível observar, pela área azul, que existe um aumento gradual dessa métrica de 0% até 20%, no horário 18:28:45. Mantendo-se próximo à 20% até o horário 18:29:30, quando começa diminuir até chegar em 0%. Esses números fazem sentido levando em consideração o plano testes, o qual aumenta e diminui gradualmente o número de requisições.

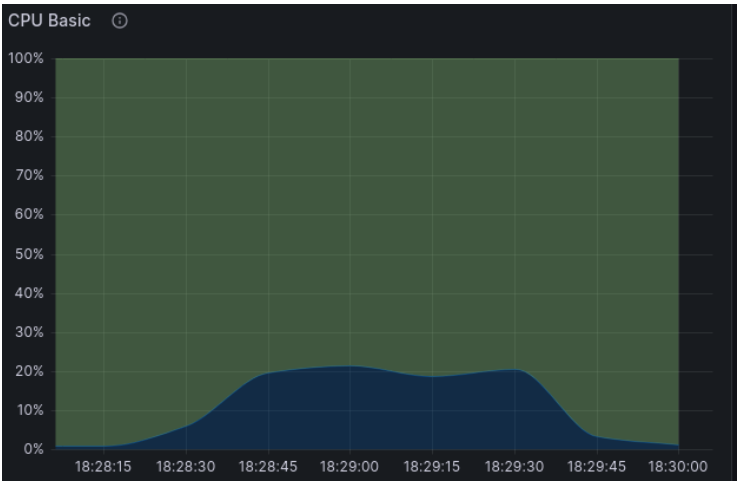

Figura 4.6 – Gráfico do uso de CPU.

Fonte: (próprios Autores)

Em relação ao uso de memória, representado pela Figura 4.7, é possível observar pela área azul do gráfico que também há um aumento gradativo no uso da memória a partir do horário 18:28:30, chegando a aproximadamente 900 MiB (*Mebibyte*) de memória, quase 50% da capacidade total, mantendo-se assim até o final do teste. Além disso, o gráfico apresenta padrão de funcionamento normal, sem apresentar indícios de vazamentos de memória.

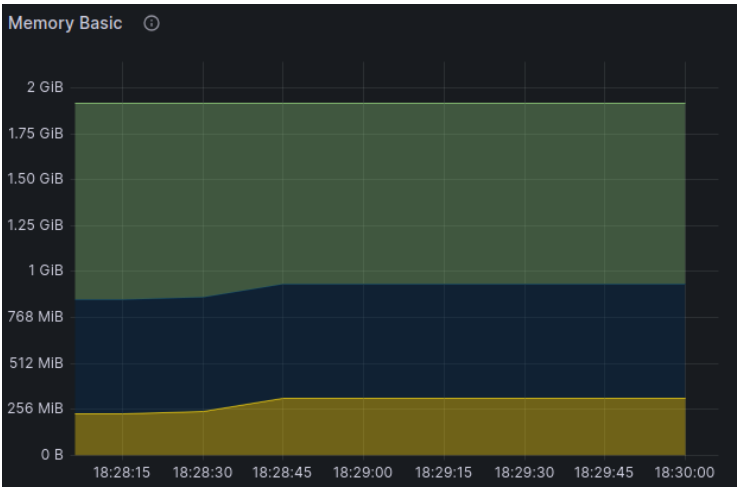

Figura 4.7 – Gráfico do uso de memória.

Fonte: (próprios Autores)

Nos resultados do tráfego de rede, notou-se que o tráfego de dados transmitido, representado pela cor azul no gráfico da Figura 4.8, chegou a um valor máximo de 2.75 Mb/s (Megabit por segundo), no horário 18:28:45. Esse valor é 11 vezes maior que o tráfego de dados recebido, indicado pela cor verde. Esses valores fazem sentido considerando que o servidor precisa enviar o resultado da aplicação a cada requisição.

Figura 4.8 – Gráfico do tráfego de rede.

Fonte: (próprios Autores)

Finalmente, foi observado que nos 3 gráficos existem pontos em comum que ocorrem as variações, principalmente comparando o uso da CPU com o tráfego de rede. É interessante notar que uma ferramenta gráfica como o Grafana possibilita a visualização do uso dos recursos de maneira clara, podendo ainda identificar padrões e tendências das métricas monitoradas.

5. Considerações

Neste tutorial foi apresentado um caso de uso de um Servidor Apache executado no sistema operacional GNU/Linux Ubuntu Server 22.04.3 LTS, sendo monitorado com o uso do Prometheus e Grafana. Os recursos monitorados foram o uso de CPU, uso total de memória e tráfego de rede. Todos os passos para

instalação e configuração do ambiente de testes foram apresentados, possibilitando a replicação dos resultados obtidos.

O monitoramento das métricas se deu através da coleta e envio das métricas do servidor para o Grafana com uso do Prometheus. Além disso, foram criados *dashboards* no Grafana, onde foi possível visualizar graficamente os recursos utilizados no decorrer do tempo. Por fim, com a análise dessas métricas, é possível acompanhar o comportamento de servidor e planejar sua configuração para suportar a alta demanda, evitando que o desempenho não seja afetado e não sofra indisponibilidade.

Com isso, o objetivo de capacitar os administradores de sistemas a monitorarem o desempenho de seus servidores, prevenir potenciais problemas e garantir a continuidade operacional foi alcançado e concluído com êxito. Todos os tópicos propostos foram abordados e explicados.

Referências

ALIOSTAD. Performance series. Disponível em: <https://byterot.blogspot.com/2013/03/performance-series-memory-leak-part-1.html>. Acesso em: 15 nov. 2023.

BEHARA, S. Cloud Native Monitoring with Prometheus. Disponível em: <https://samirbehara.com/2019/05/30/cloud-native-

monitoring-with-prometheus/>. Acesso em: 15 nov. 2023.

BRAZIL, B. Prometheus : up & running: infrastructure and application performance monitoring. Sebastopol, Ca: O'reilly Media, 2018. Acesso em 24 out. 2023.

CANONICAL. About the Ubuntu project | Ubuntu. Disponível em: <https://ubuntu.com/about>. Acesso em: 23 out. 2023.

CANONICAL. Download Ubuntu Server | Download | Ubuntu. Disponível em: <https://ubuntu.com/download/server>. Acesso em: 23 out. 2023.

CANONICAL. Install and Configure Apache. Disponível em: <https://ubuntu.com/tutorials/install-and-configure-apache#1-overview>. Acesso em: 16 nov. 2023.

CANONICAL. Introduction to virtualisation. Disponível em: <https://ubuntu.com/server/docs/virtualization-introduction>. Acesso em: 23 out. 2023.

CANONICAL. Libvirt. Disponível em: <https://ubuntu.com/server/docs/virtualization-libvirt>. Acesso em: 23 out. 2023.

CANONICAL. Screen-by-screen installer guide. Disponível em: <https://ubuntu.com/server/docs/install/step-by-step>. Acesso em: 30 out. 2023.

CANONICAL. Virtualization - qemu. Disponível em: <https://ubuntu.com/server/docs/virtualization-qemu>. Acesso em: 23 out. 2023.

CARVALHO, André C. P. L. F de; LORENA, Ana C. Introdução à Computação - Hardware, Software e Dados. Grupo GEN, 2016. E-book. ISBN 9788521633167. Disponível em: https://app.minhabiblioteca.com.br/#/books/9788521633167/. Acesso em: 27 out. 2023.

CAWLEY, C. Ubuntu Desktop vs. Ubuntu Server: What's the Difference? Disponível em: < https://www.makeuseof.com/tag/difference-ubuntu-desktop-ubuntu-server/ >. Acesso em: 7 nov. 2023

COCK, J. Reducing Energy & Water Consumption in Industry 4.0 with Grafana. Disponível em: <https://www.factry.io/blog/reducing-energy-water-consumption-in-industry-4.0-with-grafana/>. Acesso em: 18 nov. 2023.

FOSTER, E. Compare 8 tools for IT monitoring. TechTarget. Disponível em: < https://www.techtarget.com/searchitoperations/feature/Compare-8-tools-for-IT-monitoring >. Acesso em: 04 nov. 2023.

DAM, J. Grafana Labs at 5: How We Got Here and Where We're Going. Grafana Labs, 4 out. 2019. Disponível em <https://grafana.com/blog/2019/10/04/grafana-

labs-at-5-how-we-got-here-and-where-were-going/>. Acesso em: 30 out. 2023.

FRISVOLD, J. An introduction to Prometheus metrics and performance monitoring. Disponível em: <https://www.redhat.com/sysadmin/introduction-prometheus-metrics-and-performance-monitoring>. Acesso em: 18 nov. 2023.

GRAFANA LABS. Download Grafana. Disponível em: <https://grafana.com/grafana/download>. Acesso em: 16 nov. 2023.

GRAFANA LABS. Grafana Features. Disponível em: <https://grafana.com/grafana/?plcmt=footer>. Acesso em: 30 out. 2023.

GRAFANA LABS. Introduction to Grafana. Disponível em: <https://grafana.com/docs/grafana/latest/introduction/>. Acesso em: 30 out. 2023.

GRAFANA LABS. Manage dashboards. Disponível em: <https://grafana.com/docs/grafana/latest/dashboards/manage-dashboards/>. Acesso em: 16 nov. 2023.

GRAFANA LABS. Results output. Disponível em: <https://k6.io/docs/get-started/results-output/>. Acesso em: 16 nov. 2023.

GRAFANA LABS. Release notes. Disponível em: <https://grafana.com/docs/grafana/latest/release-notes/>. Acesso em: 3 nov. 2023.

GRAFANA LABS. Start the Grafana server. Disponível em: <https://grafana.com/docs/grafana/latest/setup-grafana/start-restart-grafana/#start-the-grafana-server-with-systemd>. Acesso em: 16 nov. 2023.

GRAFANA LABS. What is Load Testing? How to create a Load Test in k6. Disponível em: <https://k6.io/docs/test-types/load-testing/>. Acesso em: 7 nov. 2023.

IBM. IBM Cloud Docs. Disponível em: <https://cloud.ibm.com/docs/monitoring?topic=monitoring-prometheus-exporters>. Acesso em: 18 nov. 2023.

IBM. IBM Documentation. Disponível em: <https://www.ibm.com/docs/pt-br/instana-observability/current?topic=instana-building-custom-dashboards>. Acesso em: 19 nov. 2023.

JETBRAINS. Tutorial: Find a memory leak. Disponível em: <https://www.jetbrains.com/help/idea/tutorial-find-a-memory-leak.html>. Acesso em: 16 nov. 2023.

KVM contributors. Disponível em: <https://www.linux-kvm.org/page/Main_Page>. Acesso em: 23 out. 2023.

ÖDEGAARD, T. The (Mostly) Complete History of Grafana UX. Grafana Labs, 2 set. 2019. Disponível em: <https://grafana.com/blog/2019/09/03/the-mostly-complete-history-of-grafana-ux/>. Acesso em: 30 2023.

OHTA, S. Obtaining the Knowledge of a Server Performance from Non-Intrusively Measurable Metrics. International Journal of Engineering and Technology Innovation, [S. l.], v. 6, n. 2, p. 135–151, 2016. Disponível em: https://ojs.imeti.org/index.php/IJETI/article/view/122. Acesso em: 26 out. 2023.

PROMETHEUS. Grafana. Disponível em: <https://prometheus.io/docs/visualization/grafana/#using>. Acesso em: 16 nov. 2023.

PROMETHEUS. Monitoring Linux host metrics with the Node Exporter. Disponível em: <https://prometheus.io/docs/guides/node-exporter/>. Acesso em: 16 nov. 2023.

PROMETHEUS. Overview | Prometheus. Disponível em: <https://prometheus.io/docs/introduction/overview>. Acesso em: 24 out. 2023.

RENITA, J.; ELIZABETH, N. E. Network's server monitoring and analysis using Nagios. 2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET). Anais...IEEE, mar. 2017b. Disponível em: http://dx.doi.org/10.1109/wispnet.2017.8300092. Acesso em: 27 out. 2023.

RUAN, Zhenyuan; SCHWARZKOPF, Malte; AGUILERA, Marcos K.; BELAY, Adam. AIFM: High-Performance, Application-Integrated Far

Memory. In: 14th USENIX Symposium on Operating Systems Design and Implementation, 2020. Acesso em: 14 nov. 2023.

SANTOSO, S.; FORD LUMBAN GAOL; MATSUO, T. The Development of Monitoring Java Web Application Server Resource Usage using JavaMelody Promotheus Grafana for Sever Maintenance. Research Square (Research Square), 7 jun. 2022. Acesso em: 30 out. 2023.

SO-IN, Chakchai. The Survey of Network Traffic Monitoring and Analysis Tools. Disponível em: <https://www.researchgate.net/profile/Chakchai-So-In/publication/241752391_A_Survey_of_Network_Traffic_Monitoring_and_Analysis_Tools/links/60cad67592851ca3aca73ca8/A-Survey-of-Network-Traffic-Monitoring-and-Analysis-Tools.pdf>. Acesso em: 14 nov. 2023.

THE APACHE SOFTWARE FOUNDATION. Welcome! Disponível em: <https://httpd.apache.org/>. Acesso em: 16 nov. 2023.

THE LINUX FOUNDATION. What is Linux? Disponível em: https://www.linux.com/what-is-linux/. Acesso em: 27 out. 2023.

THE PHP GROUP. PHP: Instalação. Disponível em: <https://www.php.net/manual/pt_BR/gmp.installation.php>. Acesso em: 16 nov. 2023.

TOMMCDON. Debug a memory leak tutorial - .NET. Disponível em: <https://learn.microsoft.com/en-us/dotnet/core/diagnostics/debug-memory-leak>. Acesso em: 16 nov. 2023.