PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA E DE ARTES
GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

VIRTUALIZAÇÃO DE SERVIDORES UTILIZANDO PROXMOX APLICADO AO AMBIENTE DEVOPS

GOIÂNIA
2023

JOÃO VICTOR PEREIRA VALENTE

VIRTUALIZAÇÃO DE SERVIDORES UTILIZANDO PROXMOX APLICADO AO AMBIENTE DEVOPS.

Trabalho de Conclusão de Curso apresentado à Escola Politécnica da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para obtenção do título de Bacharel em Engenharia de Computação.

Orientador:Prof.M.E.E. Rafael Leal Martins

GOIÂNIA

2023

VIRTUALIZAÇÃO DE SERVIDORES UTILIZANDO PROXMOX APLICADO AO AMBIENTE DEVOPS.

Trabalho de Conclusão de Curso aprovado em sua forma final pela Escola Politécnica, da Pontifícia Universidade Católica de Goiás, para obtenção do título de Bacharel em Engenharia de Computação, em _____/_____/_____.

_____

Prof. Ma. Ludmilla Reis Pinheiro dos Santos

Coordenadora de Trabalho de Conclusão de Curso

Banca examinadora:

_____

Orientador: Prof. M.E.E. Rafael Leal Martins

_____

Prof. M.E.E. Fernando Gonçalves Abadia

_____

Prof. M.E.E. Angélica Da Silva Nunes

DEDICATÓRIA

A minha família, que sempre esteve comigo, especialmente ao meu pai que sempre me apoiou e esteve ao meu lado.

AGRADECIMENTOS

A Deus por todas as coisas.

Aos meus pais pelo apoio e companheirismo, e por serem o meu exemplo e espelho em tudo que faço.

Ao meu pai, meu grande amigo.

Ao meu orientador Rafael Leal Martins, pela grande ajuda na construção deste trabalho, sempre solícito e disponível para ajudar.

RESUMO

A virtualização de servidores é uma solução prática e eficiente para a criação de novos serviços em redes de computadores. Dentre as diversas ferramentas disponíveis para a criação e gerenciamento desses servidores, destaca-se o Proxmox. Esta monografia tem como objetivo apresentar as características e possibilidades dessa ferramenta, analisando sua capacidade de criar servidores de forma virtual, com base nos princípios de confiabilidade, desempenho e usabilidade. Além disso, serão implementadas algumas aplicações DevOps no ambiente virtual, a fim de testar sua funcionalidade e desempenho. Com isso, busca-se explorar o potencial do Proxmox como uma solução viável e eficaz para a virtualização de servidores, proporcionando uma infraestrutura ágil e flexível para atender às demandas crescentes das redes de computadores.

Palavras-chave: Virtualização de servidores.Proxmox.DevOps.

ABSTRACT

Server virtualization is a practical and efficient solution for creating new services in computer networks. Among the various tools available for creating and managing these servers, Proxmox stands out. This thesis aims to present the characteristics and possibilities of this tool, analyzing its ability to create virtual servers based on the principles of reliability, performance, and usability. Additionally, some DevOps applications will be implemented in the virtual environment to test their functionality and performance. The goal is to explore the potential of Proxmox as a viable and effective solution for server virtualization, providing an agile and flexible infrastructure to meet the growing demands of computer networks.

Keywords: Server virtualization.Proxmox.DevOps.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS

Me(a) .....Mestre(a)

Prof. .....Professor

TCC .....Trabalho de Conclusão de Curso

VM……Virtual Machine

DEV…….Desenvolvimento

OPS…….Operações

KVM…….Máquina Baseada em kernel

SO……Sistema Operacional

LXC……..Linux Containers

Qemu…...Quick Emulator

SPARC…Scalable Processor Architecture

ARM…….Advanced RISC Machine

GUI ……..Graphical User Interface

NFS …….Network File System

SUMÁRIO

1.INTRODUÇÃO

A virtualização de servidores tem desempenhado um papel fundamental na otimização e no gerenciamento eficiente dos recursos de hardware em ambientes corporativos. Com o avanço das tecnologias de virtualização, as organizações podem maximizar a utilização de seus servidores físicos, reduzir os custos de infraestrutura e simplificar a gestão de seus ambientes de TI. Nesse contexto, surge o Proxmox, uma solução de virtualização de código aberto que tem ganhado destaque no mercado devido à sua flexibilidade e recursos avançados.

O presente trabalho de conclusão de curso (TCC) tem como objetivo principal explorar a virtualização de servidores utilizando o Proxmox e sua aplicação no ambiente DevOps. O termo DevOps refere-se a uma cultura e conjunto de práticas que promovem a colaboração entre as equipes de desenvolvimento (Dev) e operações (Ops), visando acelerar o processo de entrega de software e melhorar a qualidade dos serviços oferecidos. A virtualização de servidores, combinada com o ambiente DevOps, oferece uma infraestrutura ágil e escalável para suportar o ciclo de vida completo de um aplicativo, desde o desenvolvimento até a produção.

Diante desse contexto, os objetivos gerais deste trabalho são:

- Explorar os fundamentos teóricos da virtualização de servidores, compreendendo conceitos, tecnologias e benefícios associados a essa abordagem.
- Investigar as características e recursos oferecidos pelo Proxmox como plataforma de virtualização de servidores.
- Analisar a cultura e as práticas do DevOps, compreendendo seus princípios e como podem ser aplicados na infraestrutura virtualizada.
- Avaliar a integração do Proxmox com ferramentas e práticas DevOps, identificando os benefícios proporcionados por essa combinação.
- Realizar um estudo de caso prático para demonstrar a aplicação do Proxmox no ambiente DevOps, destacando os resultados obtidos e os desafios enfrentados.

Com base nos objetivos gerais, os objetivos específicos deste trabalho são:

- Revisar a literatura existente sobre virtualização de servidores, Proxmox e ambiente DevOps, identificando as melhores práticas e os estudos de caso relevantes.
- Descrever em detalhes as características e funcionalidades do Proxmox, destacando suas vantagens e possíveis limitações.
- Investigar as práticas e ferramentas do DevOps, enfatizando sua aplicação na infraestrutura virtualizada e os benefícios resultantes.
- Realizar um levantamento das principais integrações e ferramentas disponíveis para integrar o Proxmox com as práticas DevOps.
- Implementar um ambiente de teste utilizando o Proxmox e práticas DevOps, realizando a avaliação dos resultados e a identificação de desafios encontrados.

Ao final deste estudo, espera-se obter um panorama claro sobre a virtualização de servidores utilizando o Proxmox no contexto do ambiente DevOps. Pretende-se identificar os benefícios dessa combinação, bem como os desafios e as melhores práticas para sua implementação efetiva. Com isso, busca-se contribuir para a disseminação do conhecimento e o uso adequado dessas tecnologias, auxiliando as organizações na otimização de seus recursos de infraestrutura e no aprimoramento de seus processos de desenvolvimento e entrega de software.

2.REFERENCIAL TEÓRICO

A fim de fornecer um embasamento adequado para a compreensão dos objetivos desta monografia, é fundamental abordar alguns conceitos básicos que envolvem as temáticas centrais deste trabalho: a virtualização. Os próximos tópicos tem como objetivo detalhar e esclarecer a estrutura e o funcionamento desse conceito, culminando no objetivo principal a ser alcançado neste trabalho.

2.1 Virtualização

A ideia embrionária de máquina virtual teve origem em uma publicação de um artigo nomeado “Time sharing processing in large fast computers”, que foi publicado na conferência internacional de processamento da informação, realizada em Nova York em 1959 (STRACHEY, 2022). O artigo foi desenvolvido pelo cientista Christopher Strachey, o conteúdo do texto trata sobre multiprogramação em tempo compartilhado, possibilitando que servidores pudessem aproveitar com mais eficiência os recursos de hardware, usando esse artigo como referência,o MIT(Massachusetts Institute of Technology - Instituto de Tecnologia de Massachusetts), desenvolveu o Compatible time sharing system (CTSS), que foi utilizado por vários fabricantes(VERAS,2015).

O CTSS foi evoluindo e a IBM adotou o conceito de multiprocessamento em mainframes, que tem como objetivo principal, permitir que várias CPUs trabalhem simultaneamente como se fossem apenas um, com isso surgiu o conceito de memória virtual que poderia ser utilizado como parte de um sistema operacional.Com isso a possibilidade de abstrair e mapear memória real para a virtual, definindo as primeiras formas de virtualização. (VERAS,2015)

A máquina virtual de processo foi um conceito inicialmente adotado que consistia em uma aplicação que era executada em um sistema operacional A, poderia ter seu comportamento emulado em um sistema operacional B, esse sistema poderia emular também processadores.(VERAS,2015)

O processo de virtualização consiste em particionar um dispositivo físico, incluindo servidores. O que anteriormente era exclusivamente um equipamento físico, pode se tornar hospedeiro de várias máquinas virtuais, esse processo é

realizado por um aplicativo. Cada equipamento tem a capacidade de executar seus sistemas operacionais independentemente. (VMware, 2022)

O aplicativo responsável por criar e executar as máquinas virtuais(VMs) é o hipervisor, possibilitando que um computador denominado como host, preste suporte a várias VMs guest, compartilhar recursos como por exemplo memória ram, armazenamento, processamento, podendo ser alocados e realocados com praticidade tanto para máquinas virtuais vigentes ou futuras.(VMware, 2022)

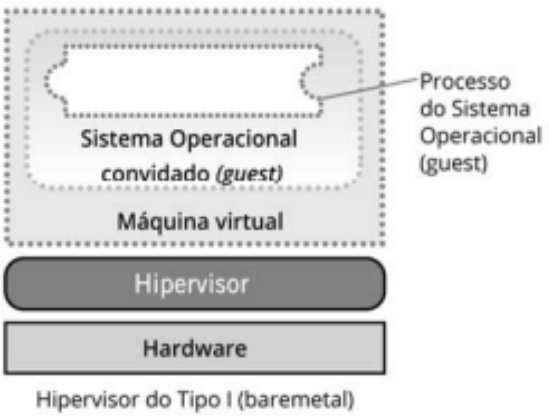
Os hipervisores podem ser classificados conforme descrito a seguir.

2.1.1 Tipos de hipervisor

O hipervisor de tipo 1 conhecido como bare-metal ou nativo, sua execução acontece diretamente no hardware, controlando o hardware e a acessibilidade do sistema operacional convidado, o objetivo desse hipervisor é garantir com que o compartilhamento de recursos de hardware entre as máquinas virtuais ali presentes, fazendo com que cada uma pense que os recursos são exclusividades de cada uma. No mercado possui alguns exemplos de hipervisor bare-metal são: VMware ESX Server,Microsoft Hyper-v, Xen server, dentre outros.(Veras,2015)

A imagem a seguir descreve a estrutura de um hipervisor bare metal:

Figura 1: Hipervisor bare metal



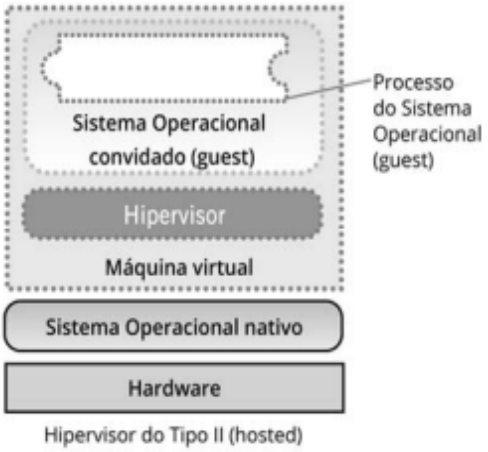
Fonte:(VERAS,2015)

Por sua vez o hipervisor do tipo 2 conhecido como hosted,fornece um ambiente de execução para outras aplicações,sendo executado sobre um sistema

operacional denominado como nativo como se pertencesse a um processo do mesmo.A camada onde ocorre a virtualização é construída por um sistema operacional hóspede e um hardware virtual, que são criados como recursos físicos que são viabilizados pelo sistema operacional (S.O) nativo.No mercado são oferecidos algumas opções como VMware player,Virtualbox e VirtualPC.(VERAS, 2015)

A Figura 2 descreve a estrutura de um hipervisor nativo, utilizando camadas identificando cada componente.

Figura 2: Hipervisor hosted



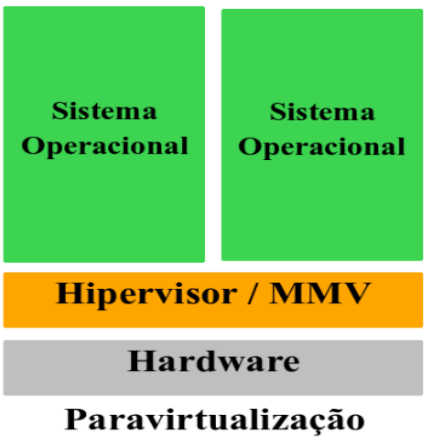
Fonte:(VERAS,2015)

A virtualização tem algumas técnicas,dentre elas a paravirtualização tem o objetivo de fugir de problemas de desempenho e a subutilização de recursos deixando claro para o sistema operacional convidado, que o mesmo está sendo executado por um hipervisor, e não no próprio hardware real da máquina. Para que isso possa acontecer o sistema operacional sofre alterações, que incluem não realizar chamadas de sistemas nativos, utilizando o hipervisor para realizar essa tarefa.(VERAS, 2015)

Os hipervisores que dispõem da paravirtualização, suas máquinas virtuais utilizam os drivers do dispositivo físico sendo controlados pelo próprio hipervisor otimizando o desempenho,um aplicativo que utiliza a paravirtualização é o Xen Open source, virtualizando tanto processador e quanto memória. (VERAS,2015)

A Figura a seguir mostra a estrutura da paravirtualização.

Figura 3: Modelo de Paravirtualização

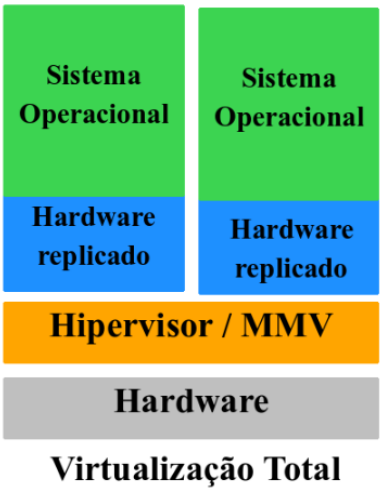


Fonte: https://www.portalgsti.com.br/2016/11/virtualizacao-completa-e-paravirtualizacao.html

Outra técnica que é importante ressaltar é conhecida como virtualização total, que consiste em abstrair completamente o sistema físico para a criação de um ambiente virtual completo. Com isso não é necessário fazer qualquer tipo de modificação no sistema operacional ou na aplicação que está sendo executada dentro dessa técnica. Essa estratégia é um grande facilitador quando se trata de migração de servidores físicos para virtuais, pois garante a total independência da aplicação e dos recursos físicos do servidor.(VERAS, 2015)

A Figura 4 demonstra a estrutura da virtualização total.

Figura 4: Modelo de virtualização total



Fonte:https://www.portalgsti.com.br/2016/11/virtualizacao-completa-e-paravirtualizacao.html

2.2 Proxmox

O Proxmox é um a aplicação com o propósito de oferecer solução para o gerenciamento de servidores virtuais, o programa possui código aberto, e utiliza de base ferramentas como QEMU/KVM e Linux Containers (LXC). Além de gerenciar servidores virtuais, é possível criar contêineres,clusters como alta disponibilidade, armazenamento e oferece um sistema de gerenciamento intuitivo acoplado em uma interface web que pode ser acessada em qualquer browser.(Proxmox,2022)

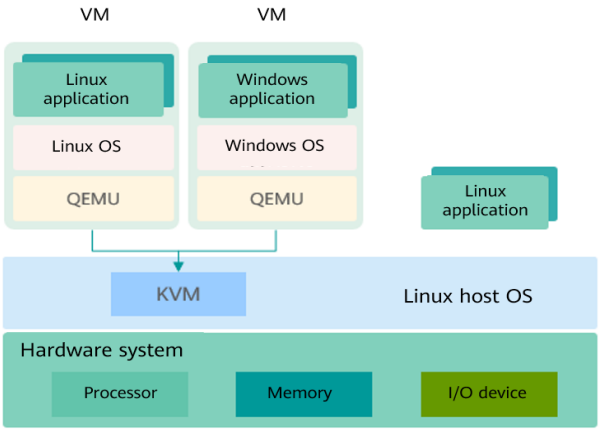
O proxmox utiliza Máquina Baseada em kernel (KVM), que foi incorporado ao linux, surgindo a possibilidade de criar máquinas virtuais isoladas e independentes. Importante ressaltar que as KVMs não dependem do sistema operacional host, porém é exigido a ativação do mesmo na BIOS.O Proxmox utiliza e proporciona um ambiente estável para máquinas virtuais baseadas em KVM.(AHMED,2017)

O funcionamento do KVM faz a conversão do sistema operacional Linux para um hipervisor bare-metal, outros tipos de hipervisores necessitam de recursos do sistema operacional para fazer ações como gerenciar memória, agendar processo e stack de entrada/saída, por sua vez o KVM possui todos esses recursos, pois faz parte do kernel do linux. A implementação das máquinas virtuais é dada como um processo regular sendo programado por um agendador linux padrão.(RedHat,2022).

Para fazer a implementação do KVM, é preciso ter um sistema linux compatível com a tecnologia, é necessário ser implementada em uma arquitetura x86, que possa oferecer suporte a virtualização. Se todos os critérios forem atendidos, é necessário apenas carregar os módulos, um emulador, e um driver que fará o auxílio para execução de sistemas adicionais.(RedHat,2022)

Na imagem a seguir, descreve a estrutura do KVM:

Figura 5: Estrutura KVM

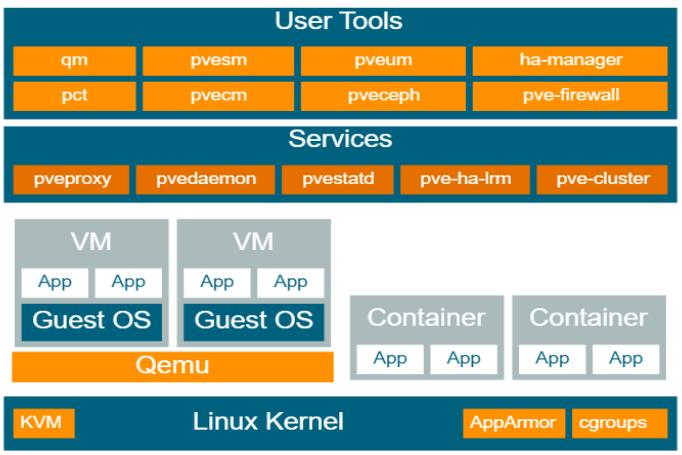


Fonte:https://docs.openeuler.org/en/docs/20.09/docs/Virtualization/introduction-to-virtualization.html

Qemu (Quick Emulator), é um hipervisor de opensource com o propósito de emular máquinas físicas. Para o host, o QEMU é um programa que tem o acesso a vários recursos físicos que por sua vez são transferidos para um computador que está sendo emulado, e o mesmo considera esses recursos como se fossem reais.(Proxmox,2022)

O Qemu pode simular diversos tipos de hardware desde ARM (Advanced RISC Machine) a Sparc (Scalable Processor Architecture), porém o Proxmox possui foco em emular apenas máquinas de 32 e 64 bits, garantindo assim abrangência na maioria dos servidores encontrados no mercado. O mesmo é utilizado como suporte das extensões do processador de virtualização através do KVM, sendo utilizados no Proxmox de forma alternada. A execução do Qemu no Proxmox é caracterizado como root.(Proxmox,2022).Na Figura 6 identifica-se a estrutura do Proxmox.

Figura 6: Estrutura em camada Proxmox



Fonte:https://pve.proxmox.com/pve-docs/pve-admin-guide.html

2.2.1 Alta disponibilidade

O Proxmox possui recursos conhecido como High Availability (HA), que proporciona alta disponibilidade entre os clusters, isso possibilita que um host físico falhar, as máquinas virtuais ali presentes serão migradas para outro host, garantindo a redundância necessária na infraestrutura. O aplicativo possui uma ferramenta integrada conhecida como ha-manager, que faz a migração de forma automática, é capaz de detectar erros e fazer failover.(Proxmox,2022).

Para isso ser possível é preciso denominar os recursos que devem ser conFigurados, posteriormente o ha-manager observa a funcionalidade correta fazendo o tratamento correto para o failover para outro host, não preciso acontecer uma falha necessariamente, a utilização do ha-manager pode ser por conveniência do gerenciador dos cluster, dependendo da demanda.(Proxmox,2022).

A matemática pode representar a disponibilidade através da probabilidade, definindo a disponibilidade como em razão de (A), que seria o tempo que um serviço pode ser utilizado, em um intervalo de tempo (B), que na maioria das vezes é determinada como porcentagem de atividade dentro de um ano. O ha-manager possui um poder de detectar erros e failovers em cerca de 2 minutos, conseguindo no máximo 99,999% de disponibilidade.(Proxmox.2022).

A Figura 7 tem o propósito de mostrar a relação de disponibilidade em um período de 1 ano.

Figura 7: Disponibilidade em um intervalo de 1 ano.

Disponibilidade %	Tempo de inatividade por ano
99	3,65 dias
99,9	8,76 horas
99,99	52,56 minutos
99.999	5,26 minutos
99,9999	31,5 segundos
99.99999	3,15 segundos

Fonte:Capturado pelo autor de Promox

Existem alguns requisitos para que possa acontecer a alta disponibilidade de forma adequada, são necessários:

- No mínimo 3 nós de cluster.
- Armazenamento compartilhado para as VMs e contêineres.
- Redundância de hardware.
- Usar componentes de “servidor” confiáveis.
- Watchdog de software.
- Dispositivos de cerca de hardware opcionais.

2.2.2 Backup

Backup é uma prática crítica em um ambiente de TI, o Proxmox VE possui uma versão integrada, fazendo o uso dos recursos de cada armazenamento e cada tipo específico de sistema convidado. O usuário administrador tem o papel de escolher o modo de consistência do backup e o tempo de inatividade do sistema convidado. Vale ressaltar que os backups são feitos de forma completa contendo a conFiguração da máquina virtual e também dos dados, os backups tem duas formas de serem inicializados pela GUI (Graphical User Interface) ou através de linha de comando vzdump.(PROXMOX,2022)

Deve ser definido antes de qualquer backup, o armazenamento onde será executada a prática. Os backups são armazenados como arquivos regulares, pode ser criado um servidor NFS (Network File System), onde o cliente pode acessar os arquivos como se fossem locais.(PROXMOX,2022).

O backup pode ser agendado, em um horário específico, a ação será iniciada automaticamente, para os nós e máquinas virtuais que foram selecionadas, isso tudo é feito no datacenter na interface gráfica do usuário, gerando uma entrada de trabalho */etc/pve/jobs.cfg,* que por sua vez deve ser validada e executada pelo daemon *pvescheduler*, usando o eventos de calendário para definir a programação.(PROXMOX,2022).

Existem alguns modos de backup com características diferentes e propósitos diferentes, que podem ser utilizados de acordo com as necessidades do usuário,dependendo do tipo de convidado, são eles modo de parada, modo de suspensão e modo instantâneo.(PROXMOX,2022)

No modo de parada, possui um nível de consistência elevado, gastando um período de inatividade na máquina virtual que não é muito alto. Entretanto o procedimento desabilita a VM, posteriormente é iniciado um processo em segundo plano que no qual é encarregado de inicializar o backup.(PROXMOX,2022).

No modo de suspensão, o propósito desse modo é prover compatibilidade, suspendendo a VM, posteriormente chama o modo instantâneo,porém ocasiona um tempo de inatividade das máquinas virtuais, e a consistência dos dados se mantém inalterada.(PROXMOX,2022)

O modo instantâneo, é o mais interessante pela redução de tempo de inatividade e dos perigos de inconsistência é inferior comparado com os modos anteriores. Nesse modo, as máquinas virtuais não são suspensas, as atividades continuam inalteradas e ficam em produção enquanto os blocos de dados são copiados.(PROXMOX,2022)

Uma ferramenta também muito utilizada para recuperação é o Snapshot, quando efetuada, faz uma cópia do estado do servidor virtual, criando um outro exatamente com as mesmas conFigurações e conteúdo, é uma prática comum para serviços que são frequentemente atualizados, e que sofrem mudanças dentro do cluster.

Promox também possui uma versão dedicada exclusivamente para backup, conhecida como Proxmox Backup Server, possuindo o foco em restauração de VMs, containers, nós físicos além de do próprio backup, é uma ferramenta que pode ser aliada ao cluster promovendo assim um menor tempo de interatividade e a manutenção de dados e a segurança dos mesmos.

2.2.3 Gestão Proxmox

O Proxmox possui uma interface web onde não é necessário efetuar um download de uma ferramenta separadamente, a interface pode ser acessada de qualquer browser, podendo ser utilizado qualquer nó do cluster para poder acessar a GUI, cada nó tem a capacidade de gerenciar todos os outros, não há necessidade de promover um nó para acesso.

A o proxmox possui a porta 8006 como padrão para efetuar a conexão, juntamente deve ser inserido o identificador do nó que será utilizado para o acesso.

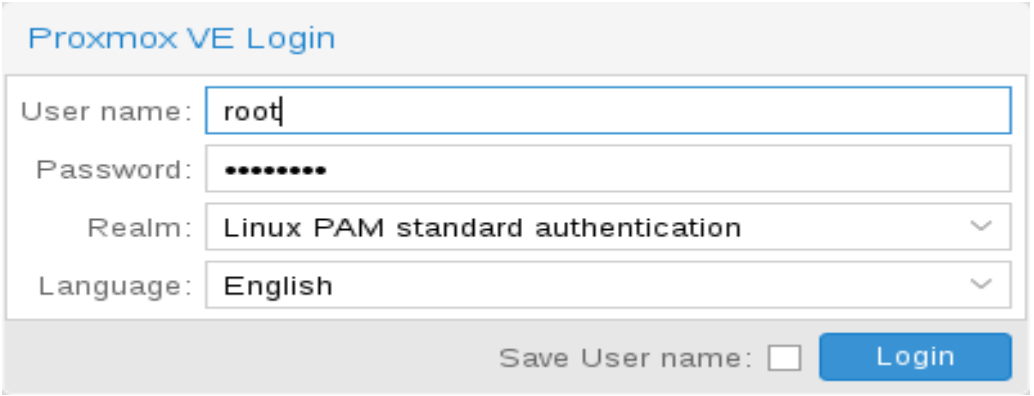
Para poder acessar definitivamente a plataforma de administração deve ser inserida a senha de acesso determinada na instalação do Proxmox.(PROXMOX,2022)

Abaixo visualiza-se algumas características da GUI.

- Integração e gerenciamento dos clusters
- Tecnologias AJAX para atualização de recursos dinamicamente
- Acesso seguro
- Interface para gerenciar um grande de VMs
- Tecnologias HTML5 ou SPICE
- Utiliza autenticação de dois fatores

As imagens e as descrições abaixo possuem o objetivo de familiarizar o ambiente Proxmox aos leitores, descrevendo a interface e algumas ferramentas essenciais que a mesma possui. A Figura 8 exemplifica a tela de autenticação, necessária para a promoção ao gerenciamento dos clusters.

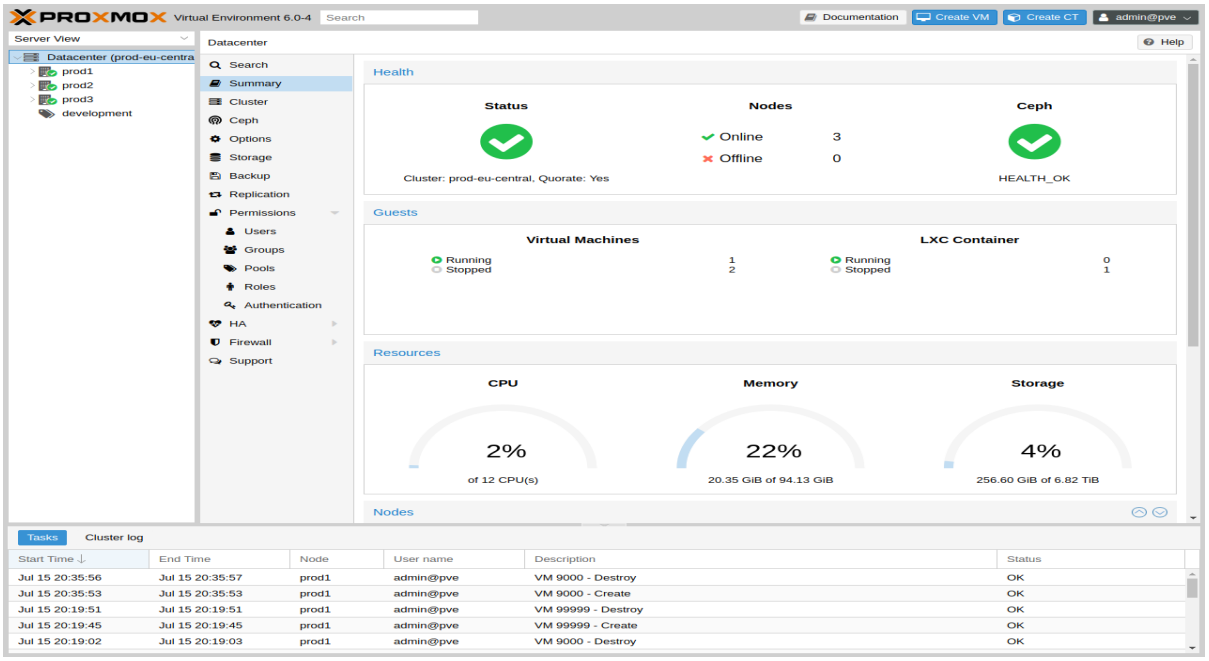
Figura 8: Tela de login Proxmox



Fonte:https://pve.proxmox.com/wiki/Graphical_User_Interface

Logo após o processo de autenticação será concluído com sucesso. A Figura 9 demonstra o menu do aplicativo, onde pode ser feito qualquer alteração,verificação, pertencentes ao cluster.

Figura 9: Menu do Proxmox



Fonte: https://pve.proxmox.com/wiki/Graphical_User_Interface

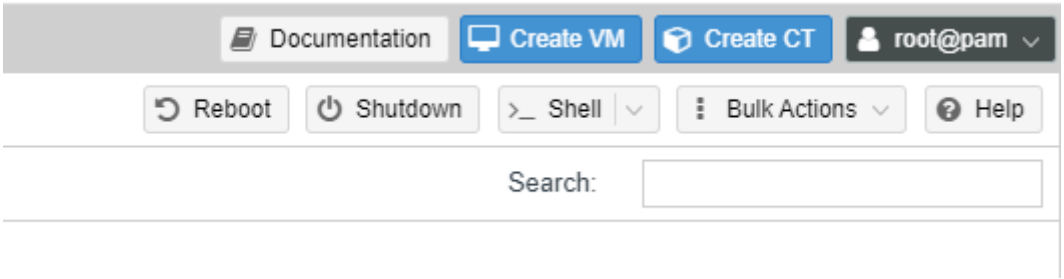
Nessa tela evidencia alguns recursos importantes, são eles Datacenter, Nodes,Guests, Storage e Pools, abaixo serão descritas as funcionalidades dos recursos mais importantes, com o objetivo de familiarizar o ambiente.

Datacenter onde é configurado todo o cluster, além de conseguir todas as informações do mesmo. Encontra-se em cada recurso sub-abas onde pode ser realizada ações ou coleta de informações, por possuir dashboard é um ponto positivo para a visualização do usuário, como pode ser observado na Figura 8.

Todas as tarefas que são executadas dentro do cluster são armazenadas em uma aba chamada Task, possuindo um histórico de atividades realizadas, dentro do ambiente Proxmox, incluindo data e hora do início e do final de cada tarefa, o usuário que fez determinada atividade, descrição da atividade que está sendo feita, e o status da atividade,

Em Nodes pode ser gerenciar os nós físicos do cluster, todos individualmente,dentro de cada nó pode ser executada ações como reiniciar, desligar, acesso ao Shell, Bulk Actions e ajuda, além de acessar cada servidor virtual pertencente ao mesmo. O acesso ao shell é feito na maioria das vezes via VNC, e não há necessidade de ir fisicamente aos servidores. A Figura 10 exemplifica os nodes.

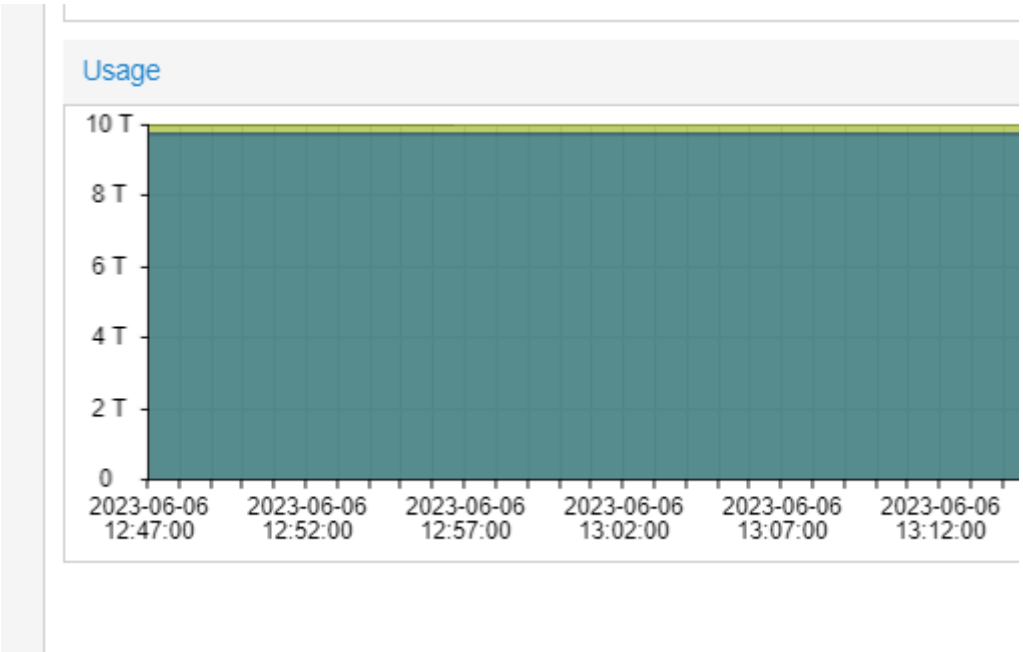
Figura 10: Nodes



Fonte:Capturado pelo autor de Proxmox

Em storage possibilita a visualização do armazenamento disponível em cada nó do cluster, pode-se inserir imagens ISO, efetuar backups, visualizar o status do storage, o armazenamento de uma máquina virtual pode ter uma mobilidade alta, pode ser alocado em um nó, ou pode ser criado um servidor somente para storage, onde pode ser gerenciado de acordo com a necessidade das VMs. Na Figura 11, ocorre a representação de um storage, e as informações necessárias para o gerenciamento do mesmo. O verde escuro indica o armazenamento ocupado e o claro o livre.

Figura 11: Storage



Fonte:Capturado pelo autor de Proxmox

2.2.4 Pontos negativos

Em um ambiente empresarial, é importante a disponibilidade, da equipe de suporte da ferramenta para possíveis dúvidas ou aparecimento de erros ou bugs que podem aparecer com o decorrer do tempo este é o principal ponto negativo da ferramenta trata-se do suporte ao usuário, como a mesma é uma ferramenta open, o suporte é feito pela própria comunidade, muitas informações são encontradas no fórum próprio do proxmox, porém são alimentadas pelos próprios usuários.

2.3 DEVOPS

DevOps é uma metodologia que surgiu como uma resposta aos desafios de comunicação e colaboração entre as equipes de desenvolvimento e segurança de software. O principal objetivo do DevOps é quebrar as barreiras entre essas áreas e promover a integração de pessoas, processos e ferramentas.

Uma das principais características do DevOps é a automatização. As ferramentas de automatização são utilizadas para automatizar tarefas repetitivas e manuais, como a compilação de código, testes, implantação e monitoramento. Essas ferramentas auxiliam na redução de erros, aumento da eficiência e aceleração do tempo de entrega.

Outra característica importante do DevOps é a cultura de colaboração e comunicação. As equipes de desenvolvimento e segurança precisam trabalhar juntas em todas as etapas do ciclo de vida do software, desde o planejamento até a manutenção.

O DevOps também enfatiza a responsabilidade compartilhada. Em vez de atribuir a responsabilidade por problemas a uma equipe específica, as equipes de desenvolvimento e segurança trabalham juntas para identificar a causa raiz dos problemas e implementar soluções.

Além disso, o DevOps promove a escalabilidade e resiliência do software. As equipes de desenvolvimento e segurança precisam trabalhar juntas para garantir que o software possa lidar com um grande volume de tráfego e se recuperar rapidamente de falhas.

A implementação do DeVOps pode trazer muitos benefícios para as empresas, incluindo a melhoria na qualidade do software, aceleração do tempo de entrega, redução de custos e melhoria na eficiência das equipes. A automatização e a colaboração constante ajudam a garantir que o software seja testado e implantado de forma mais rápida e precisa, reduzindo erros e falhas. Além disso, a automatização ajuda a reduzir custos, já que menos tempo é gasto em tarefas manuais e repetitivas, permitindo que as equipes se concentrem em tarefas de maior valor.

3. ESTUDO DE CASO

A seguir será descrito o caso de uso, onde o Proxmox será utilizado para implementar um ambiente com algumas aplicações DevOps, o objetivo é demonstrar a capacidade do hypervisor em prover serviços com qualidade, e descrever as vantagens de um ambiente DevOps. Nesse caso foram escolhidas algumas aplicações específicas sendo elas Jenkins,Docker,Ansible,Redmine e Kubernetes.

3.1 Jenkins

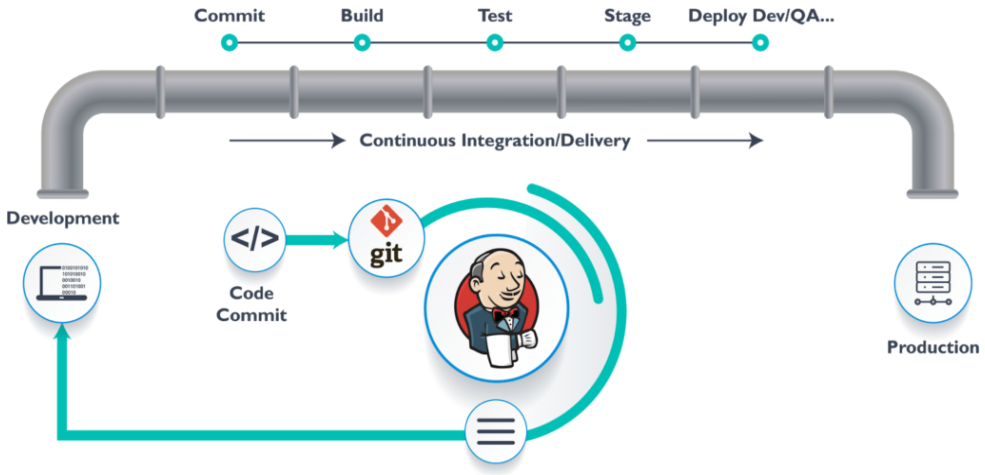
Jenkins trata-se de uma ferramenta de automação de código aberto que tem se tornado cada vez mais popular entre os profissionais de desenvolvimento de software.É uma ferramenta flexível e personalizável, permitindo que os usuários configurem pipelines de entrega contínua que atendam às necessidades específicas de suas equipes de desenvolvimento.

Uma vantagem do Jenkins é que ele pode ser facilmente integrado a outras ferramentas de desenvolvimento, como Git, Docker e Kubernetes. Isso permite que os desenvolvedores possam automatizar todo o processo de construção, teste e implantação de seu software, garantindo maior eficiência e qualidade na entrega do produto final.

Vale ressaltar que o Jenkins tem a capacidade de notificar automaticamente os membros da equipe quando ocorrer um erro ou quando uma nova versão do software estiver disponível para implantação.Os usuários podem visualizar o progresso do pipeline em tempo real, o que permite identificar rapidamente quaisquer problemas e corrigi-los antes que eles se tornem críticos.

O Jenkins possui uma comunidade ativa de usuários e desenvolvedores, pois possuem vários adeptos em todo mundo. Como uma ferramenta de código aberto como o Proxmox, o Jenkins é mantido por uma grande comunidade global de usuários que estão constantemente trabalhando em novos recursos e melhorias para a plataforma. Na Figura 12 conseguimos visualizar o ciclo de integração e entrega contínua dentro do Jenkins.

Figura 12: Entrega e Integração contínua



Fonte:https://blog.syncitgroup.com/jenkins-architecture-and-pipeline/

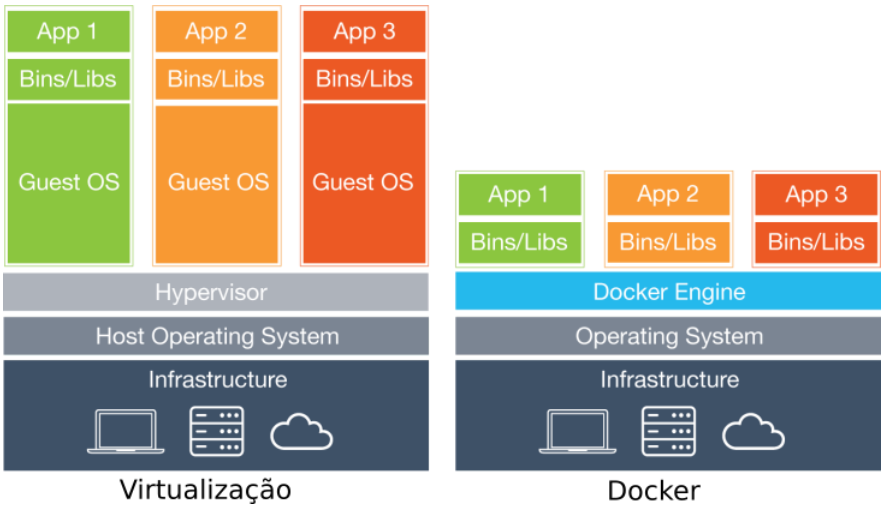
3.2 Docker

Docker é uma tecnologia de contêiner que tem se tornado cada vez mais popular entre os profissionais de TI. Ela permite que os desenvolvedores criem, empacotem e distribuam aplicativos de maneira mais rápida e eficiente do que com as estruturas tradicionais de infraestrutura.

Uma das principais vantagens do Docker é sua capacidade de criar contêineres independentes e isolados que podem ser executados em qualquer ambiente, independentemente do sistema operacional ou das configurações da infraestrutura. Isso significa que os desenvolvedores podem criar aplicativos uma vez e depois implantá-los em qualquer lugar, sem precisar se preocupar com incompatibilidades ou erros de configuração.

Vale apontar que no Docker é sua eficiência em termos de recursos. Ao contrário das estruturas tradicionais de infraestrutura, que requerem que os aplicativos sejam executados em máquinas virtuais completas, o Docker permite que várias instâncias de um aplicativo sejam executadas em um único host, compartilhando os recursos da máquina de maneira mais eficiente. Isso significa que os desenvolvedores podem economizar dinheiro em hardware e custos de infraestrutura, além de reduzir o tempo de implantação do aplicativo, isso pode ser visualizado na Figura 13.

Figura 13: Estrutura docker



Fonte: https://docker-unleashed.readthedocs.io/aula1.html

O Docker oferece maior segurança e flexibilidade do que as estruturas tradicionais de infraestrutura. Como os contêineres são independentes e isolados, os desenvolvedores podem implantar diferentes versões do mesmo aplicativo em diferentes contêineres sem se preocupar com conflitos ou riscos de segurança. Isso significa que os desenvolvedores podem atualizar ou testar diferentes versões do aplicativo sem afetar a versão em produção.

Vale ressaltar é a facilidade de uso e gerenciamento. Com o Docker, os desenvolvedores podem criar, empacotar e distribuir aplicativos de maneira rápida e fácil, usando ferramentas familiares como o Git e o Docker Hub. Além disso, o Docker oferece recursos de monitoramento e gerenciamento integrados, permitindo que os desenvolvedores monitorem e gerenciem seus aplicativos de maneira eficiente.

3.3 Ansible

Ansible é um software de automação de TI que pode ajudar a simplificar e gerenciar infraestruturas complexas. É uma ferramenta poderosa que permite automatizar a configuração, a implantação e a orquestração de aplicativos e serviços em uma variedade de ambientes.

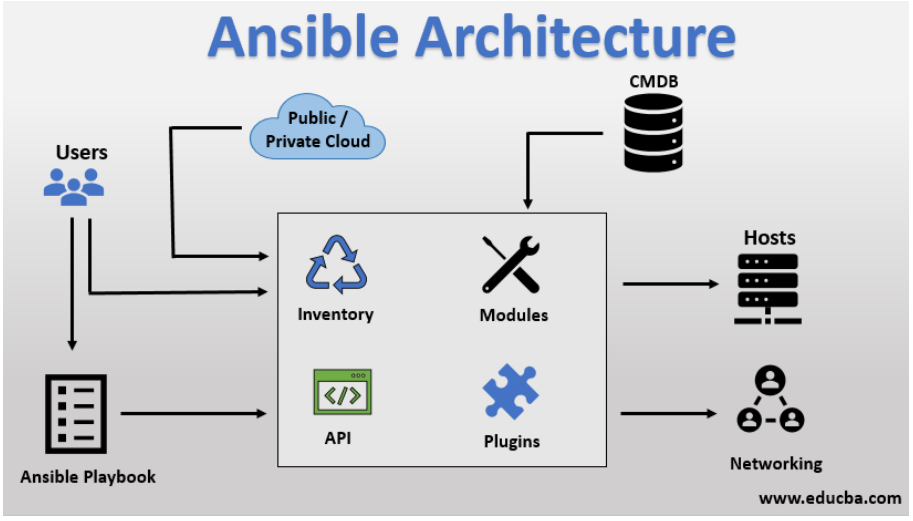
Uma das principais vantagens do Ansible é sua simplicidade. Ele usa uma linguagem de configuração simples, que é fácil de entender e usar, mesmo para aqueles que não têm experiência em programação. Além disso, o Ansible não requer a instalação de agentes em servidores remotos, o que facilita muito a configuração.

Outra vantagem do Ansible é sua capacidade de executar tarefas em paralelo em vários servidores ao mesmo tempo. Isso significa que você pode implantar rapidamente aplicativos em grandes conjuntos de servidores, economizando tempo e esforço. O Ansible também permite que você gerencie configurações em vários ambientes, como produção, teste e desenvolvimento.

O Ansible também é altamente seguro. Ele usa uma abordagem baseada em SSH para se comunicar com os servidores, o que significa que as informações confidenciais, como senhas e chaves SSH, são criptografadas durante a transmissão. Além disso, o Ansible permite o uso de autenticação baseada em certificados, o que ajuda a reduzir o risco de ataques de senhas fracas.

O Ansible também possui uma grande comunidade de usuários e desenvolvedores que estão constantemente criando e compartilhando novos módulos e recursos para a ferramenta. Isso significa que você pode aproveitar uma ampla gama de recursos pré-criados para automatizar tarefas comuns de TI, além de ter suporte e soluções para problemas mais complexos.

Figura 14: Arquitetura Ansible



Fonte:https://www.educba.com/ansible-architecture/

3.4 Redmine

O Redmine é um sistema de gerenciamento de projetos e acompanhamento de problemas, altamente valorizado por sua eficiência e funcionalidades abrangentes. Com uma interface intuitiva e amigável, o Redmine oferece uma plataforma versátil para equipes de todos os tamanhos, permitindo a colaboração eficaz e a entrega bem-sucedida de projetos.

Uma das principais vantagens do Redmine é a sua capacidade de personalização. Com uma ampla gama de plugins e temas disponíveis, é possível adaptar o Redmine às necessidades específicas de cada organização. Isso permite que as equipes criem fluxos de trabalho personalizados, definem permissões de acesso e configurem painéis de controle personalizados para melhor atender aos requisitos do projeto.

Além disso, o Redmine oferece um conjunto abrangente de recursos que abrangem desde o acompanhamento de problemas e tarefas até a gestão de documentos e calendários. Com a funcionalidade de acompanhamento de problemas, os usuários podem facilmente registrar, priorizar e atribuir tarefas a membros da equipe, permitindo uma comunicação clara e uma visão geral do progresso do projeto.

Outro destaque do Redmine é o seu sistema de rastreamento de tempo. Ele permite que os usuários registrem o tempo gasto em tarefas e projetos, o que é essencial para o controle de custos, a análise de produtividade e o planejamento futuro. Essa funcionalidade fornece insights valiosos sobre como os recursos estão sendo alocados e ajuda a otimizar a utilização da equipe.

O Redmine oferece recursos avançados de relatórios e gráficos, permitindo uma análise detalhada do desempenho do projeto. Os gerentes podem gerar relatórios personalizados, visualizar o progresso do projeto em formato de Gantt e identificar áreas que requerem atenção adicional. Essa capacidade de monitoramento e análise ajuda a manter os projetos no caminho certo e a tomar decisões informadas.

O Redmine também suporta integração com outras ferramentas populares, como controle de versão, sistema de suporte ao cliente e muito mais. Essa interoperabilidade facilita a colaboração com outras equipes e simplifica o fluxo de trabalho geral.

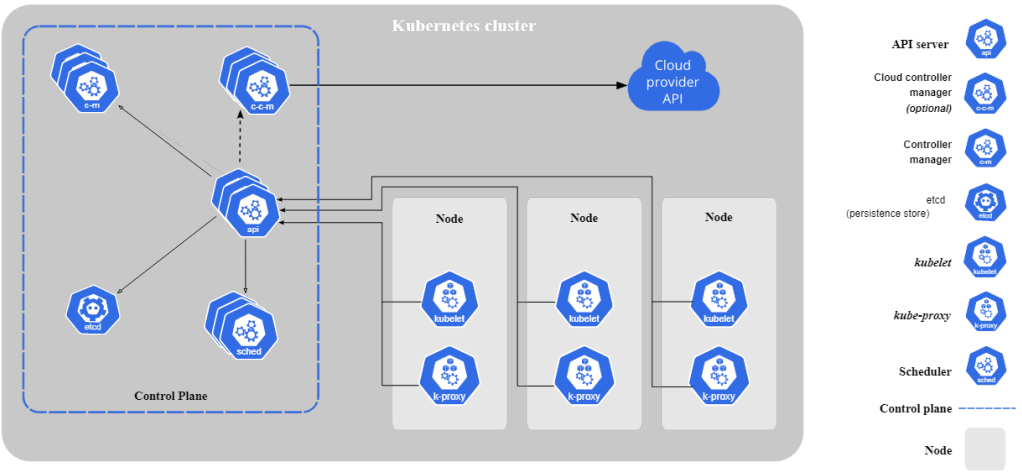
3.5 Kubernetes

Kubernetes é um software livre e de código aberto para orquestração de contêineres, sendo um dos projetos mais importantes da Cloud Native Computing Foundation (CNCF). Sua importância e uso têm crescido rapidamente devido à popularidade dos contêineres, que proporcionam maior portabilidade, eficiência e escalabilidade para aplicações em nuvem.

O Kubernetes realiza o gerenciamento de contêineres por meio da organização dos mesmos em grupos chamados de Pods, que são então distribuídos em Nós, que podem ser compostos por máquinas físicas ou virtuais, conforme a estrutura do cluster. Esses Nós são administrados pela camada de gerenciamento e contam com os recursos necessários para a execução adequada dos Pods.(Kubernetes,2023)

Um cluster Kubernetes é composto por um conjunto de servidores de processamento, conhecidos como nós, que executam aplicações em contêineres. Cada cluster possui pelo menos um servidor de processamento (nó de trabalho).Os nós de processamento hospedam os Pods, que são os componentes básicos de uma aplicação. A camada de gerenciamento é responsável por gerenciar os nós de processamento e os Pods no cluster. Em ambientes de produção, é comum que a camada de gerenciamento seja executada em vários computadores e que um cluster seja formado por vários nós, proporcionando tolerância a falhas e alta disponibilidade.(Kubernetes,2023), a Figura 15 descreve a forma que esse cluster é organizado.

Figura 15: Cluster Kubernetes



Fonte:https://kubernetes.io/pt-br/docs/concepts/overview/components/

O Kubernetes facilita a criação, o gerenciamento e a escalabilidade de contêineres em larga escala. Ele automatiza tarefas importantes, como a distribuição de contêineres em vários nós de um cluster, balanceamento de carga, gerenciamento de armazenamento, atualizações de software e recuperação de falhas. Com o Kubernetes, é possível implantar e executar aplicativos em qualquer ambiente de nuvem, incluindo nuvens públicas, privadas e híbridas.

Uma das vantagens do Kubernetes é sua capacidade de oferecer alta disponibilidade e tolerância a falhas, garantindo que as aplicações estejam sempre em funcionamento, mesmo em caso de falhas de hardware ou software. O Kubernetes também é altamente escalável, permitindo que aplicativos sejam implantados em larga escala, de forma consistente e confiável.

Vale ressaltar que o Kubernetes tem a possibilidade de automatizar tarefas repetitivas e demoradas, como o provisionamento de recursos e a configuração de ambientes de desenvolvimento, teste e produção. O Kubernetes permite a criação de pipelines de entrega contínua, garantindo que as atualizações de software sejam entregues de forma rápida e segura.

O Kubernetes é flexível e configurável, permitindo que as empresas personalizem suas implantações de acordo com suas necessidades específicas. Ele suporta uma ampla variedade de plataformas de contêineres e ferramentas de gerenciamento.

3.6 Arquivos YAML

Os arquivos YAML têm sido amplamente utilizados para representar dados de maneira estruturada e organizada, o que facilita sua leitura e escrita tanto por humanos quanto por máquinas. Sua versatilidade permite que sejam empregados em diversos contextos, desde a configuração de aplicativos até a definição de estruturas de dados complexas, bem como na transferência de informações entre sistemas.

Devido à sua sintaxe intuitiva, os arquivos YAML são acessíveis e permitem uma representação clara e concisa de informações. Sua aplicação é abrangente e vai além do desenvolvimento de software. Diversas linguagens de programação podem ler e interpretar arquivos YAML, o que os torna compatíveis com diferentes ecossistemas tecnológicos.

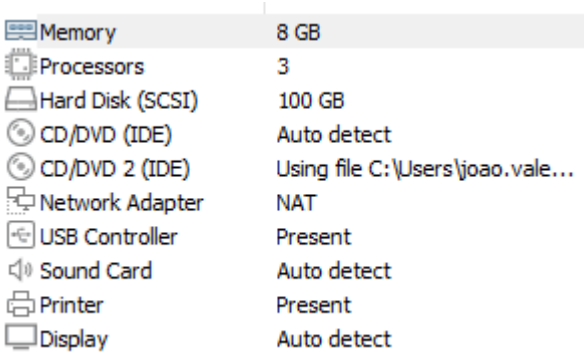
Esse formato é especialmente popular em ferramentas de automação, como Ansible, Docker Compose e Kubernetes, onde desempenham um papel fundamental na definição de configurações e especificações. Sua capacidade de estruturar dados complexos de maneira legível torna os arquivos YAML uma escolha frequente para facilitar o fluxo de informações entre sistemas e processos automatizados.

4.CENÁRIO PROPOSTO

Foi criado com o objetivo de apresentar a plataforma de forma mais prática uma implementação de um caso de uso, para explorar de forma empírica as funcionalidades do Proxmox, utilizando-o para a implementação de um ambiente DevOps. Nesse exemplo foi utilizada a versão mais recente do hipervisor a 7.2-1 em formato ISO, pode ser encontrada no site oficial do Proxmox, a máquina que foi utilizada para esse experimento foi uma Dell 5090 , possuindo 32 GB de memória ram, 500 GB de armazenamento ssd, e com um processador Intel(R) Core(TM) i7-10700M CPU @ 2.90GHz.

O Proxmox foi instalado dentro de uma máquina virtual, para isso foi utilizado o VMware que pode ser encontrado no site oficial do VMware, na Figura 16 foi colocada todas as especificações das máquinas virtuais, incluindo informações básicas das mesmas.

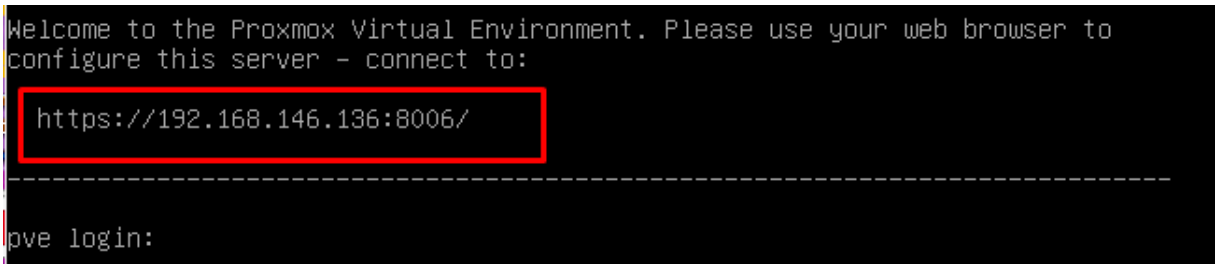
Figura 16: Especificações da VM



Fonte:Capturado pelo autor de VMware

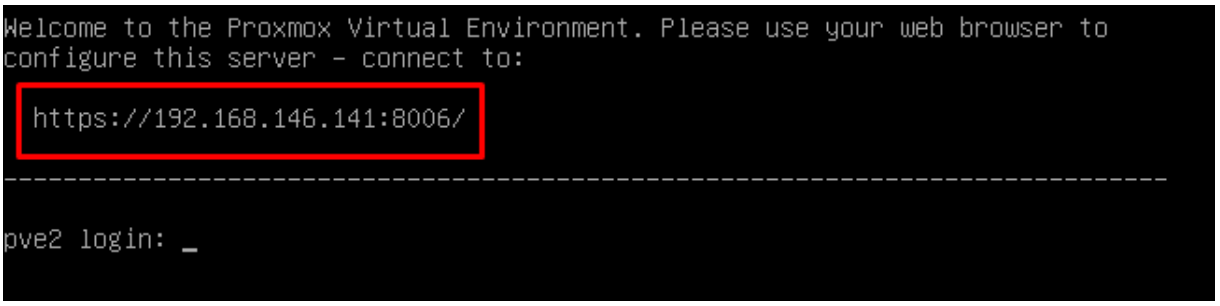
Seguindo todos os passos com sucesso da instalação do Proxmox, será liberado o acesso para a interface no navegador onde é inserido o ip e a porta nesse caso 192.168.146.136/192.168.146.141 e 8006, esse processo pode ser visualizado na Figura 17 e 18, salientado na imagem por um retângulo vermelho, em seguida na Figura 19 pode-se visualizar a interface, onde é localizada todas as informações do servidor, como explicado anteriormente neste trabalho

Figura 17: Instalação concluída



Fonte: Capturado pelo autor de Proxmox

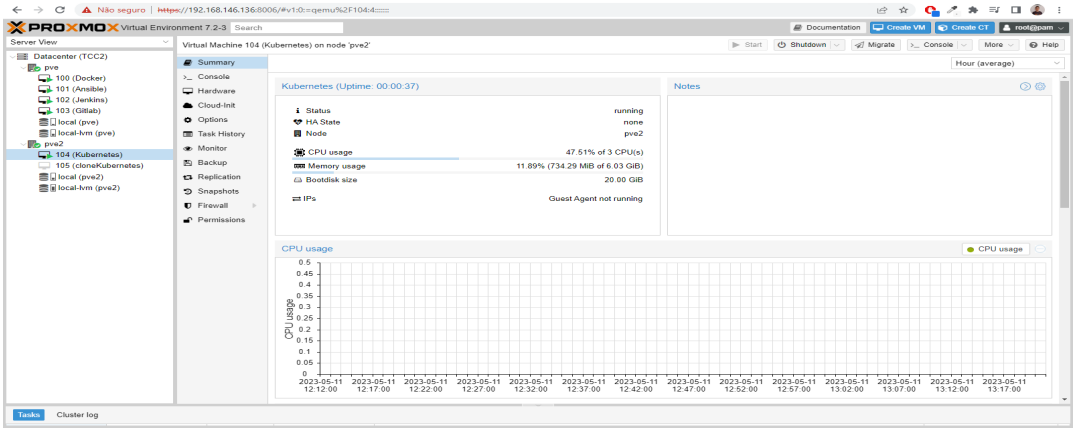
Figura 18: Instalação concluída



Fonte: Capturado pelo autor de Proxmox

Com isso podemos acessar a interface gráfica em qualquer browser,onde pode ser feita as configurações e a criação do ambiente proposto neste trabalho, como pode ser visto na Figura 14.

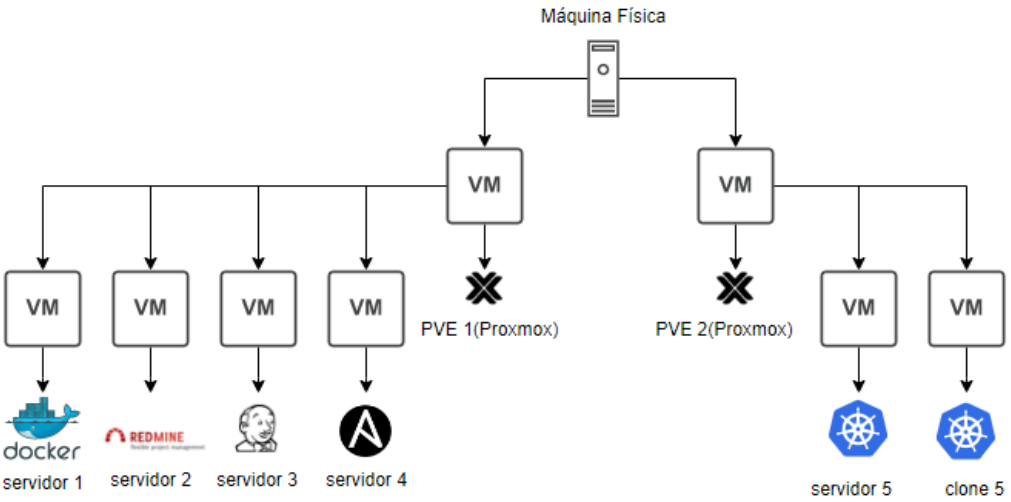
Figura 19:Acesso a interface gráfica



Fonte:Capturado pelo autor de Proxmox

Com base nisso foi elaborado um diagrama para exemplificar a arquitetura do projeto, podendo ser visualizada na Figura 20.

Figura 20: Arquitetura do projeto

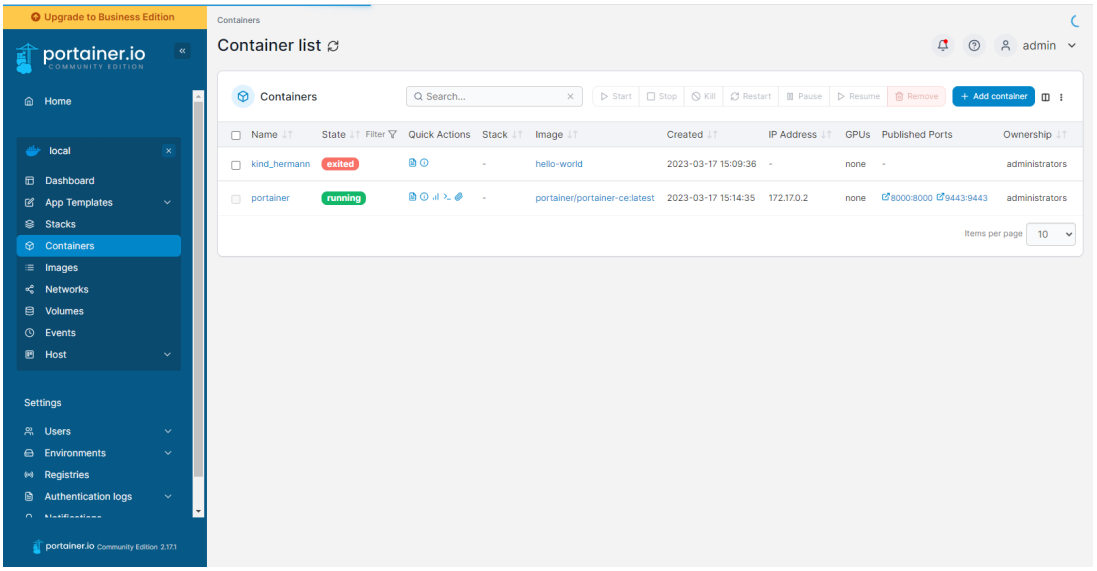


Fonte:Autoria própria

Foram criados dois servidores com objetivo de ativar a tecnologia HA (High Availability), para garantir a redundância, mantendo um ambiente com alta acessibilidade, no primeiro servidor denominado pve foram instalados 4 servidores, “Docker”,”Ansible”,”Jenkins”,”Gitlab”, no segundo servidor denominado pve2, foi instalado o servidor ”kubernetes” e um “cloneKubernetes” para ser um backup caso ocorra algum tipo de erro .Os servidores receberam o nome das aplicações que os mesmos vão hospedar.

Todos os servidores possuem instalados em comum o docker para subir as aplicações utilizando container, instalados também juntamente o Portainer.io para gerir graficamente os containers e utilizar algumas funcionalidades que serão necessárias neste trabalho , para a instalação do mesmo é necessário instalar o Docker, logo após instalar o gerenciador com o comando docker run -d -p 8000:8000 -p 9443:9443 --name portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce:latest, a instalação foi efetuada com sucesso, como pode-se notar na Figura 21.

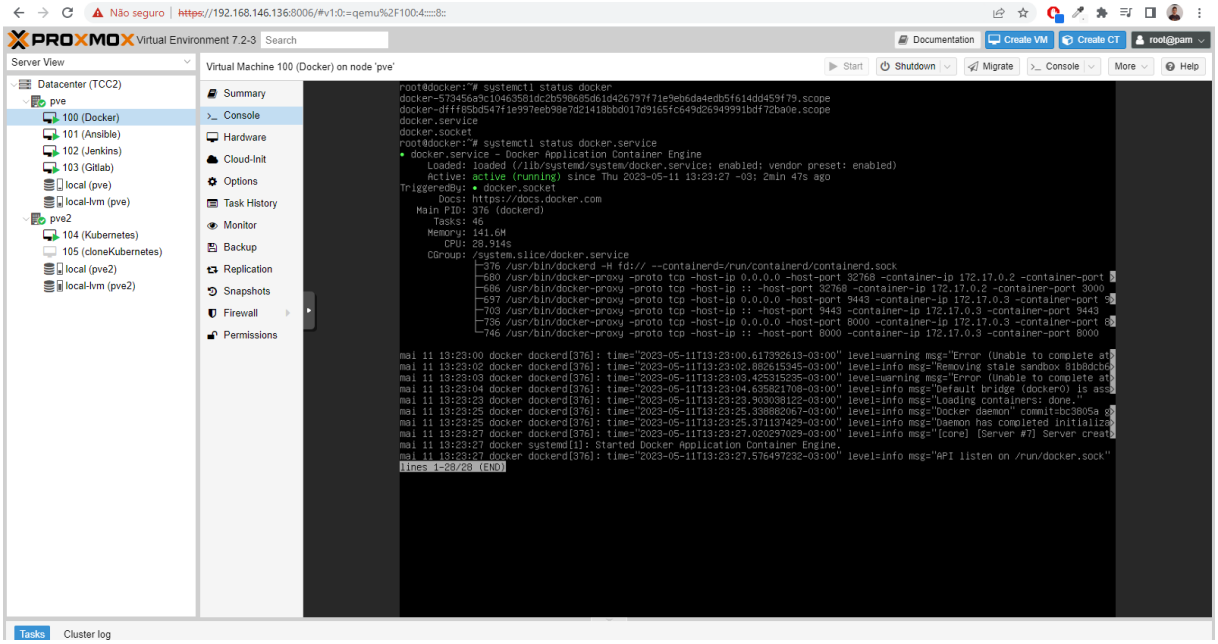
Figura 21: Instalação do porteiner.io



Fonte: Capturado pelo autor de Porteiner.io

Todos os serviços estão funcionando normalmente, as Figuras seguintes têm objetivo de demonstrar o funcionamento e aplicabilidade de cada um dentro de um ambiente devops.

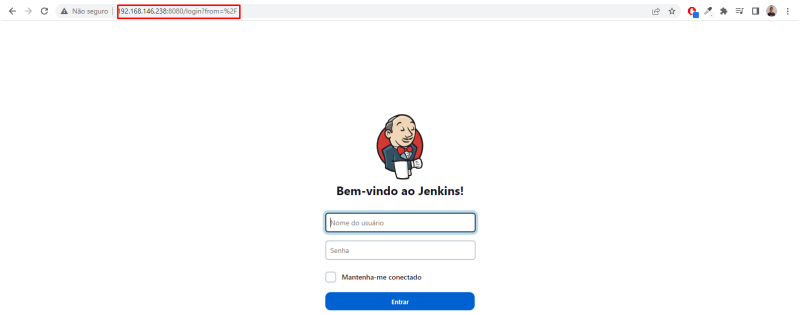
Figura 22: Serviço docker em funcionamento



Fonte:Capturado pelo autor de Proxmox

O serviço jenkins está funcionando normalmente no servidor 0.0.0.0, o mesmo permanece estável, e funcional, pronto para ser utilizado de acordo com as necessidades e demandas da empresa.

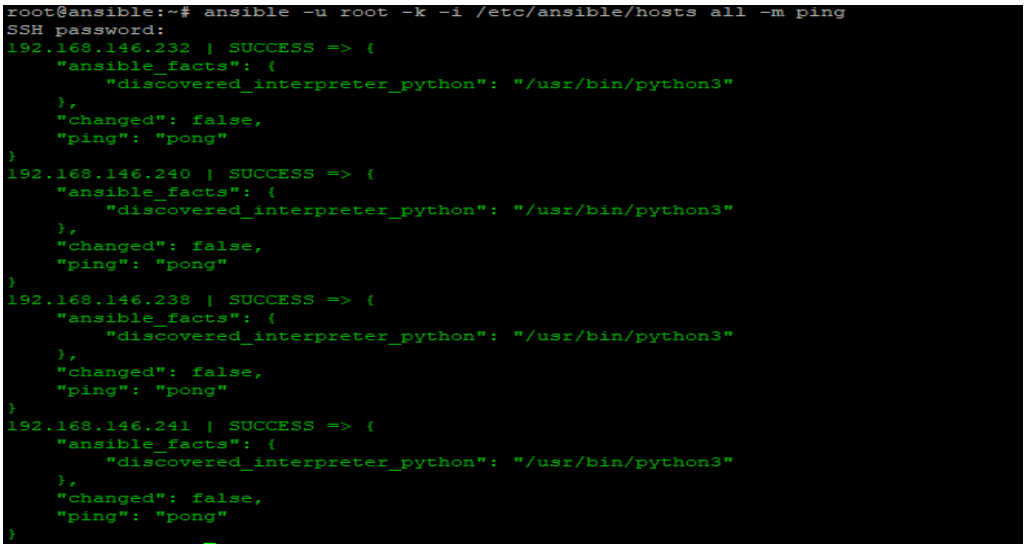
Figura 23: Serviço jenkins em funcionamento



Fonte:Capturado pelo autor de Jenkins

O Ansible está funcionando normalmente, todos os outros servidores do projeto foram colocados dentro do arquivo hosts, com isso pode-se automatizar instalações/atualizações e outras tarefas automaticamente, de forma simultânea, em todos os servidores. A Figura 24 mostra que todos servidores estão conectados de forma simultânea no servidor Ansible.

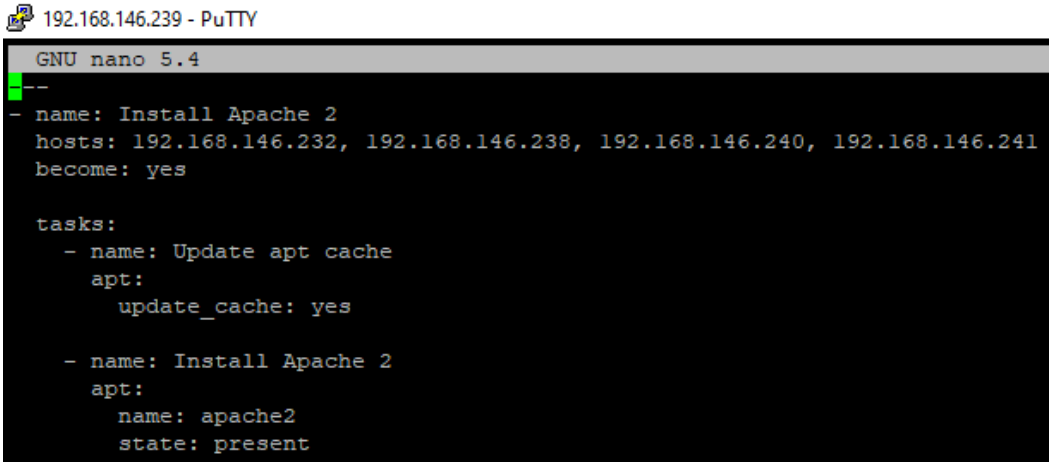
Figura 24: Ansible



Fonte: Capturado pelo autor de Debian

Foi criado um arquivo .yml denominado install_apache.yml, que faz a instalação do apache2 em todos os servidores do projeto, o arquivo pode ser visualizado a seguir:

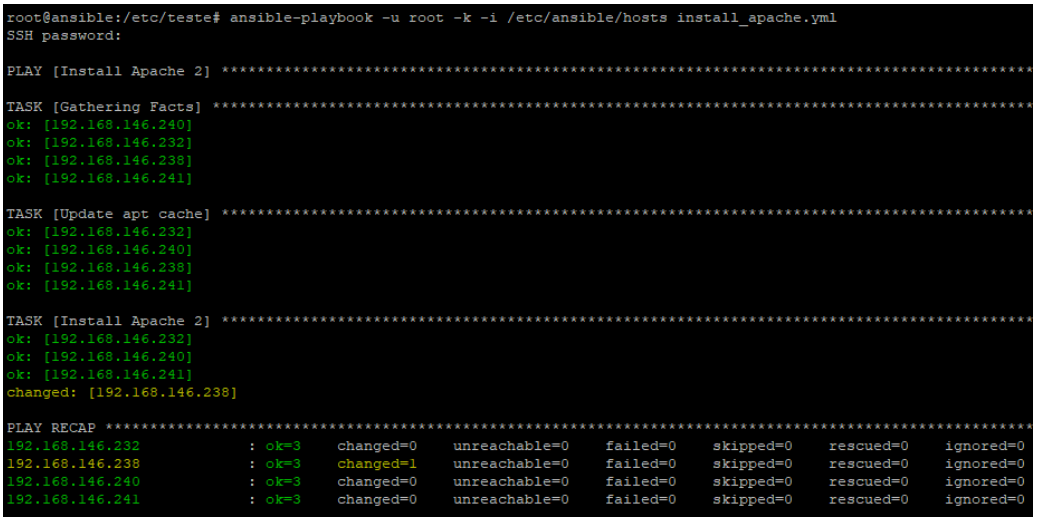
Figura 25: Arquivo .yml apache



Fonte: Capturado pelo autor de Debian

Se estiver tudo certo o processo de instalação vai começar a partir do comando ansible-playbook -u root -k -i /etc/ansible/hosts install_apache.yml, e o retorno do status quando tudo estiver correto pode ser visualizado na Figura 26.

Figura 26: Apache2 instalado em todos os servidores



Fonte:Capturado pelo autor de Debian

O Kubernetes está funcionando normalmente, cumprindo todos os requisitos de qualidade, dentro das possibilidades e limitações de hardware disponíveis. Para o teste foi implementado um projeto denominado “Kubenews” de Autoria do Fabrício Veronez, nesse arquivo possui um arquivo .yml (YAML). que está disponível publicamente, abaixo encontra-se o mesmo:

```
# Deployment do Postgre
apiVersion: apps/v1
kind: Deployment
metadata:
  name: postgre
spec:
  selector:
    matchLabels:
      app: postgre
  template:
    metadata:
      labels:
        app: postgre
    spec:
      containers:
        - name: postgre
          image: postgres:14.3
          ports:
            - containerPort: 5432
          env:
          - name: POSTGRES_PASSWORD
            value: "Kube#123"
          - name: POSTGRES_USER
            value: "kubenews"
          - name: POSTGRES_DB
            value: "kubenews"
---
apiVersion: v1
```

```
kind: Service
metadata:
  name: postgre
spec:
  selector:
    app: postgre
  ports:
  - port: 5432
    targetPort: 5432
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: kubenews
spec:
  replicas: 1
  selector:
    matchLabels:
      app: kubenews
  template:
    metadata:
      labels:
        app: kubenews
      annotations:
        prometheus.io/scrape: "true"
        prometheus.io/port: "8080"
        prometheus.io/path: "/metrics"
    spec:
      containers:
      - name: kubenews
        image: fabricioveronez/kube-news:latest
        env:
        - name: DB_DATABASE
          value: "kubenews"
```

```
        - name: DB_USERNAME
          value: "kubenews"
        - name: DB_PASSWORD
          value: "Kube#123"
        - name: DB_HOST
          value: "postgre"
---
apiVersion: v1
kind: Service
metadata:
  name: kube-news
spec:
  selector:
    app: kubenews
  ports:
  - port: 80
    targetPort: 8080
    nodePort: 30000
  type: LoadBalancer
```

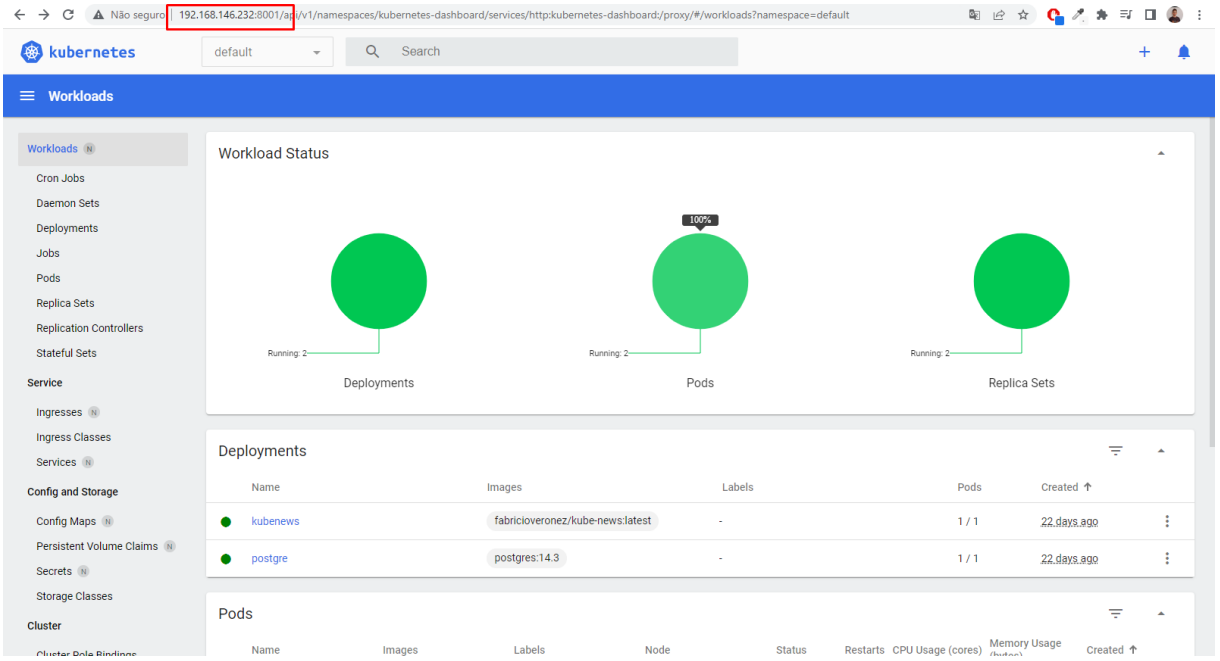
O primeiro serviço é chamado "postgre" e é associado ao deployment "postgre". Ele define um serviço que roteia o tráfego para o deployment usando o seletor "app: postgre" e expõe a porta 5432.

O segundo deployment é chamado "kubenews" e usa a imagem "fabricioveronez/kube-news:latest" do Docker Hub. Ele define um contêiner com o nome "kubenews" e também especifica variáveis de ambiente para configurar as informações do banco de dados PostgreSQL (nome do banco de dados, nome de usuário, senha e host).

O segundo serviço é chamado "kube-news" e é associado ao deployment "kubenews". Ele define um serviço que roteia o tráfego para o deployment usando o seletor "app: kubenews" e expõe a porta 80. Além disso, é definido como um tipo LoadBalancer e usa o nodePort 30000 para permitir o acesso externo ao serviço.

Nota-se na Figura 27, que toda aplicação está funcionando normalmente e atendendo as funcionalidades que propunha.

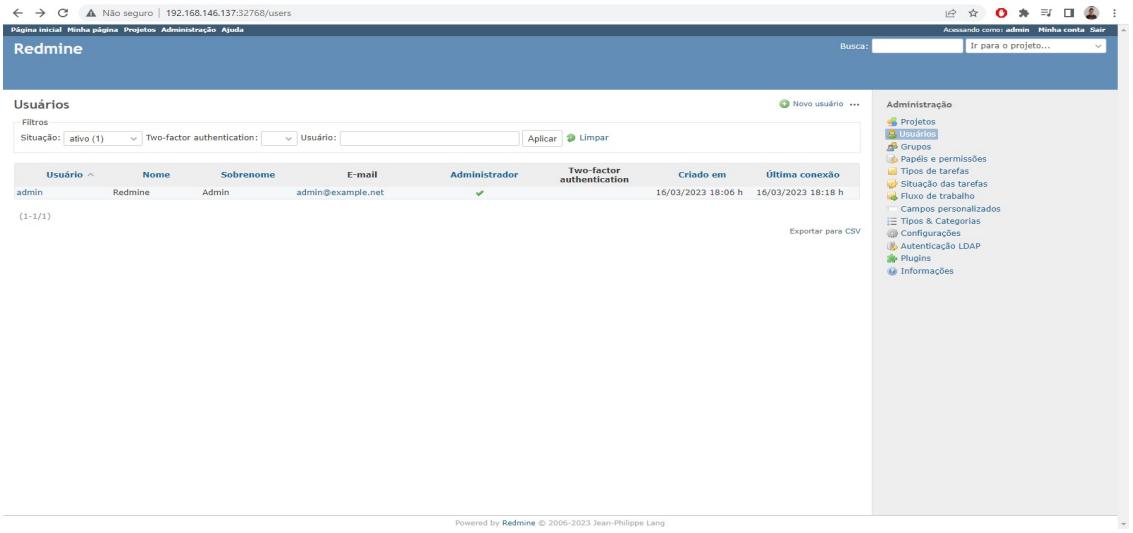
Figura 27: Serviço kubernetes



Fonte: Capturado pelo autor de Kubernetes

O Redmine está estável, com todas suas funcionalidades, sendo possível utilizá-la em um ambiente de produção sem maiores problemas,para teste foi criado um usuário administrador e tarefas,seguindo a proposta da aplicação, isso pode ser notado na Figura 28.

Figura 28: Redmine em funcionamento.



Fonte: Capturado pelo autor do Redmine.

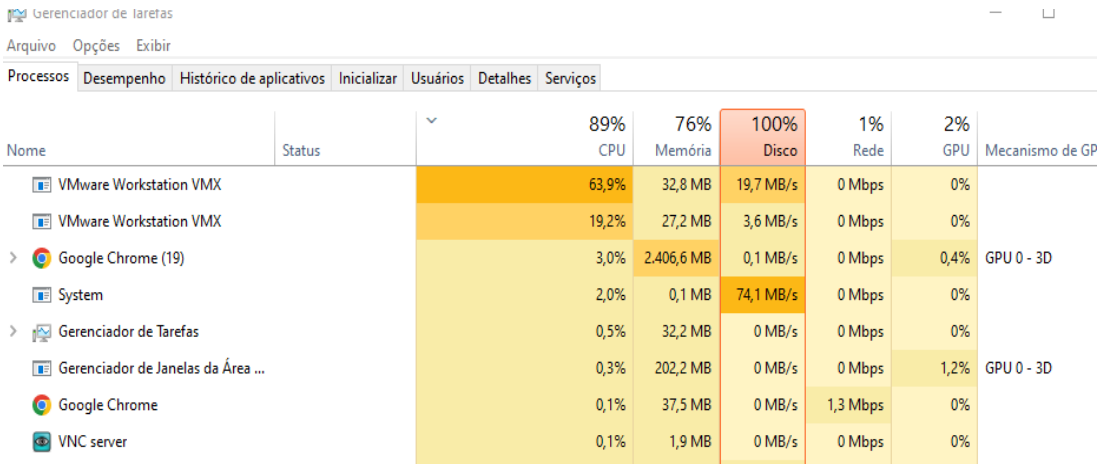
4.1 Desafios encontrados

Durante a implementação do caso de uso, uma dificuldade significativa que foi enfrentada foi o gargalo de hardware. O gargalo de hardware ocorre quando os recursos físicos disponíveis em um servidor, como capacidade de processamento, memória e armazenamento, são insuficientes para suportar adequadamente as demandas das máquinas virtuais e das práticas DevOps.

Ao implementar a virtualização de servidores, uma das expectativas é a otimização e a maximização dos recursos de hardware. No entanto, em determinados momentos, especialmente em cenários com alta demanda computacional e escalabilidade, a infraestrutura física pode se tornar um fator limitante.

Recomenda-se que para uma implementar uma infraestrutura parecida com a do caso de uso, utilizar recursos de hardware mais potentes, apesar de nenhum serviço ter sido interrompido, notou-se lentidão, isso pode ser notado na Figura 29.

Figura 29: Gargalo de Hardware



Fonte: Capturado pelo autor de Windows

5. CONSIDERAÇÕES FINAIS

Este trabalho de conclusão de curso teve como objetivo principal explorar a virtualização de servidores utilizando o Proxmox no ambiente DevOps, considerando os desafios, as soluções e os benefícios proporcionados por essa combinação. Através da análise dos fundamentos teóricos, da investigação das características do Proxmox e das práticas do DevOps, bem como da realização de um estudo de caso prático, foi possível compreender a importância e as vantagens dessa abordagem.

A virtualização de servidores tem desempenhado um papel crucial na otimização dos recursos de hardware e na simplificação da gestão de ambientes de TI. Através da criação de máquinas virtuais, é possível maximizar a utilização dos servidores físicos, reduzir os custos de infraestrutura e promover a flexibilidade no provisionamento e escalabilidade dos recursos. O Proxmox, como uma solução de virtualização de código aberto, oferece recursos avançados e uma interface intuitiva para gerenciar e controlar os ambientes virtualizados.

No contexto do ambiente DevOps, a virtualização de servidores utilizando o Proxmox possibilita uma infraestrutura ágil e escalável para suportar todo o ciclo de vida de um aplicativo, desde o desenvolvimento até a produção. A colaboração entre as equipes de desenvolvimento e operações é facilitada pela flexibilidade e consistência proporcionadas pela virtualização, permitindo a implantação rápida e repetível de ambientes de teste, a integração contínua e a entrega contínua de software.

Durante o desenvolvimento deste trabalho, foi identificada a dificuldade do gargalo de hardware, que ocorre quando os recursos físicos disponíveis são insuficientes para suportar as demandas das máquinas virtuais e das práticas DevOps. Essa dificuldade pode resultar em lentidão, instabilidade e indisponibilidade dos serviços. No entanto, foram exploradas estratégias para contornar o gargalo de hardware, como a otimização de recursos existentes, o balanceamento de carga e o dimensionamento automático.

Diante disso, é evidente que a virtualização de servidores utilizando o Proxmox no ambiente DevOps oferece benefícios significativos. Através dessa abordagem, as organizações podem alcançar maior eficiência, agilidade e escalabilidade em seus processos de desenvolvimento e entrega de software. A integração do Proxmox com as práticas DevOps permite a automatização de tarefas,

o monitoramento e a correção rápida de problemas, além de promover a colaboração e a comunicação efetiva entre as equipes.

No entanto, é importante ressaltar que a superação do gargalo de hardware e o sucesso da virtualização de servidores no ambiente DevOps exigem planejamento adequado, avaliação contínua dos requisitos de desempenho e investimento na infraestrutura física quando necessário. A correta seleção de recursos de hardware e o monitoramento constante são essenciais para garantir a efetividade e a estabilidade do ambiente virtualizado.

Por fim, este trabalho contribuiu para a disseminação do conhecimento sobre a virtualização de servidores utilizando o Proxmox no ambiente DevOps. Através da revisão da literatura, da análise das características do Proxmox, das práticas do DevOps e da realização de um estudo de caso prático, foram apresentados os benefícios, os desafios e as soluções relacionadas a essa abordagem.

Espera-se que este estudo seja útil para profissionais de TI, pesquisadores e organizações interessadas em adotar a virtualização de servidores utilizando o Proxmox no contexto do ambiente DevOps. A compreensão dos fundamentos, a aplicação das melhores práticas e a superação dos desafios contribuirão para uma infraestrutura de TI mais eficiente, ágil e escalável, impulsionando a inovação e a entrega de valor aos clientes.

REFERÊNCIAS

ANSIBLE. **Ansible Documentation**. Disponível em: **https://docs.ansible.com/**. Acesso em: 23 abr. 2023.

DEBIAN. **Debian - The Universal Operating System. Versão 11**. Disponível em: **https://www.debian.org/**. Acesso em: 22 ago. 2022.

DOCKER. **What is Docker?** Disponível em: **https://www.docker.com/what-docker.** Acesso em: 23 abr. 2023.

JENKINS. **Jenkins - The leading open source automation server**. Disponível em: **https://www.jenkins.io/**. Acesso em: 23 abr. 2023.

KUBERNETES. **Documentação oficial**. Disponível em: **https://kubernetes.io/pt-br/docs/home/**. Acesso em: 19 set. 2022.

PACKT PUBLISHING LTD. **MASTERING PROXMOX: Build virtualized environments using the Proxmox VE hypervisor.** 3. ed. Birmingham: Packt Publishing Ltd., 2017.

PORTAINER.IO. **Portainer - Simple management UI for Docker**. Versão 2.9.2. Disponível em: **https://www.portainer.io/**. Acesso em: 23 jun. 2023.

PROXMOX. **Proxmox VE**. Disponível em: **https://pve.proxmox.com/wiki/Main_Page**. Acesso em: 19 set. 2022.

RED HAT. **O que é um hipervisor?** Redhat, 2022. Disponível em: **https://www.redhat.com/pt-br/topics/virtualization/what-is-a-hypervisor#tipos-de-hipervisores.** Acesso em: 19 set. 2022.

REDMINE. Redmine - **Project Management and Issue Tracking**. Disponível em: **https://www.redmine.org/**. Acesso em: 23 abr. 2023.

STRACHEY, Christopher. **Time sharing in large fast computers**. Disponível em: **https://archive.org/details/large-fast-computers**. Acesso em: 19 set. 2022.

VERAS, M.; CARISSIMI, A. **Virtualização de Servidores**. Escola Superior de Redes, 2015, v. 2.0.0-b.

VMWARE. **Definição da virtualização de servidores**. VMware, 2022. Disponível em: **https://www.vmware.com/br/topics/glossary/content/server-virtualization.html.** Acesso em: 19 set. 2022.

VMWARE. **Virtualização: conceitos básicos**. Disponível em: **https://www.vmware.com/br/virtualization/how-does-virtualization-work.html.** Acesso em: 19 set. 2022