



PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Instituto de Ciências Exatas e de Informática

Projeto de Infraestrutura de Rede: Solo Forte Agropecuária

Bruno Alfeu Mendes de Araújo¹

Gabriel Amâncio de Oliveira²

Guilherme de Souza Mendonça Silva³

Isaac Samuel de Carvalho⁴

Matheus Godinho Blaselbauer⁵

Yan Guimarães Martins⁶

Fábio Leandro Rodrigues Cordeiro⁷

Resumo

Este trabalho insere-se no contexto de planejamento e implementação de infraestrutura de redes de computadores para a empresa fictícia de médio porte Solo Forte Agropecuária, seguindo um fluxo desde o levantamento dos requisitos aos testes de conexão, para que as filiais e a sede da empresa se comuniquem corretamente e isto resolva as dores de problemas de conexão da empresa. Para isso são utilizados os serviços de rede HTTP, FTP, DNS, NFS, DHCP, Banco de Dados com PostgreSQL, AD(Active Directory) + GPO(Group Policy Object). A justificativa para tal implementação reside na necessidade de uma infraestrutura de TI robusta que suporte a expansão da empresa, otimize a logística e centralize a gestão de dados. O objetivo deste projeto é prototipar e configurar os serviços de rede fundamentais para a operação da Solo Forte Agropecuária, conhecendo os ambientes de nuvem(cloud) e físicos(on-premise) para a hospedagem dos serviços citados.. Como resultado, obteve-se a configuração funcional de todos os serviços propostos. Conclui-se que a arquitetura de rede híbrida proposta é viável e atende aos requisitos operacionais iniciais da empresa, demonstrando a correta implementação dos serviços de rede essenciais.

Artigo apresentado ao Instituto de Ciências Exatas e Informática da Pontifícia Universidade Católica de Minas Gerais, campus Contagem, como pré-requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

1 Matheus Godinho Blaselbauer – 1497302@sga.pucminas.br

2 Guilherme de Souza Mendonça Silva – guilherme.silva.1483950@sga.pucminas.br

3 Gabriel Amâncio de Oliveira – gabriel.oliveira.1486220@sga.pucminas.br

4 Bruno Alfeu Mendes de Araújo – bruno.alfeu@sga.pucminas.br

5 Isaac Samuel de Carvalho – 1216559@sga.pucminas.br

6 Yan Guimarães Martins – 1496439@sga.pucminas.br

7 Fábio Leandro Rodrigues Cordeiro – fabio@sga.pucminas.br

Palavras-chave: Infraestrutura De redes; Serviços de rede; DHCP; Ambiente Híbrido; Virtualização.

ETAPA 1- PROTOTIPAGEM E PLANEJAMENTO

Muitas empresas possuem problemas para pensar em como desenvolver uma boa arquitetura de rede e pensar nas melhores escolhas de acordo com suas necessidades, como por exemplo o número de dispositivos necessários para estruturar a rede, a topologia que virá a ser utilizada de acordo com suas necessidades, e como permitir uma comunicação eficiente de forma que não haja nenhum defeito crítico na rede. Todo o plano para se estruturar uma rede precisa ser bem montado e escalável de acordo com as principais necessidades empresariais, desde o levantamento dos requisitos ao teste e execução das conexões. Por isso, é essencial escolher uma boa metodologia a seguir para definir esses parâmetros e permitir que o produto final seja fiel à ideia da empresa.

Portanto, uma abordagem bem consolidada no contexto de redes de computadores é a metodologia top-down, em que as decisões começam de decisões mais genéricas e migram para questões mais específicas. No contexto de arquitetura de redes, primeiro são levantados os requisitos gerais para se desenvolver a rede e posteriormente a arquitetura e estrutura lógica e física da rede são projetadas. Para se desenvolver a etapa 1 esta foi a metodologia adotada.

Assim sendo, durante a etapa 1 foi utilizado um software de simulação de redes, o Cisco Packet Tracer, fornecido pela Cisco, para estruturar a rede de acordo com uma topologia em conjunto com planilhas para enumerar a quantidade, em média, de equipamentos por categoria, e os gastos de acordo com a conectividade entre os mesmos, como a velocidade de transferência entre um dispositivo e outro, por exemplo.

Para definir estes parâmetros, foram seguidas quatro etapas de acordo com a definição da metodologia top-down:

1. Identificação das necessidades e objetivos da rede: Uma empresa de agricultura com porte de pequeno a médio com duas filiais e uma sede, com o objetivo de montar uma rede bem otimizada com o menor desperdício de gastos possível e disponibilidade de alcance para regiões mais afastadas.
2. Projeto lógico: Escolha da topologia de barramento, em que todos os computadores estão conectados por um único cabo, para estruturar a relação entre os roteadores. Mapeamento das políticas de roteamento com otimização para se obter o melhor custo-benefício na ligação por barramento. Estruturação de um diagrama de cabeamento com direcionamento dos custos estimados. Mapeamento dos endereços IPv4 das máquinas e hierarquia de distribuição dos mesmos. Interconexão entre as redes locais para permitir o roteamento

externo, utilizando cabos DTE/DCE.

3. Projeto físico: Organização do cabeamento no rack. Correlação e ordenação física das portas para conexão entre a matriz e as filiais. Distribuição entre os computadores por unidade e os tipos de conexão seguindo a relação matriz-filial.

ETAPA 2- CONFIGURAÇÃO DOS SERVIÇOS DE REDE EM NUVEM E ON-PREMISE

Muitas empresas recentemente criadas enfrentam dificuldades para disponibilizar e gerenciar servidores físicos próprios em suas instalações. O custo para comprar computadores completos geralmente é muito elevado e o gerenciamento de toda a arquitetura e disponibilidade dos serviços do zero geralmente demanda um elevado conhecimento técnico e cautela, sendo assim a empresa responsável por gerenciar toda a infraestrutura. Portanto esse setup pode não ser a opção mais rentável para startups. Daí nasce o conceito de computação em nuvem, em que provedores terceirizados oferecem serviços virtuais hospedados em servidores remotos previamente configurados e gerenciados em troca de uma mensalidade em que o usuário paga geralmente conforme demanda de uso. Ou seja, o cliente pode escalar todo o setup e infraestrutura de seus serviços conforme necessário e já utiliza uma base de infraestrutura pré-pronta, recurso usualmente chamado de IaaS ou Infraestrutura como Serviço, sendo a AWS um ótimo exemplo.

Com isso, nessa etapa os serviços de rede exigidos anteriormente na etapa 1 foram configurados, cada um, ou em ambiente de nuvem utilizando a arquitetura da AWS, ou on-premise através do software VirtualBox. Para isso, inicialmente cada aluno conheceu os conceitos básicos de configuração de cada ambiente e posteriormente os serviços de rede foram divididos entre os alunos de acordo com as seções especificadas a seguir:

1.1 Ambiente Cloud

1.1.1 Protocolo HTTP

O protocolo de Transferência de Hipertexto(HTTP), é o alicerce da comunicação de dados na web. O HTTP é um protocolo de camada de aplicação, onde opera no modelo cliente-servidor, normalmente o cliente(que geralmente é um navegador) inicia a troca através de uma requisição HTTP e o servidor responde com uma Resposta HTTP. Os responsáveis pela comunicação entre o cliente e servidor são os protocolos de comunicação(como HTTP e o TCP/IP) que define as regras para a troca de dados e a própria infraestrutura de rede(roteadores, switches e cabos) que fisicamente conecta o cliente e o servidor. Seus principais métodos de requisições são: GET, POST, PATCH, PUT e DELETE. Existem também os códigos de status, os mais comuns são: 200, 400, 403, 404 e 500. E existem os cabeçalhos, que são os metadados que são enviados entre cliente-servidor.

Neste passo a passo da configuração do HTTP, precisamos ter um servidor Ubuntu como um dos pré-requisitos. Nesse exemplo de configuração vamos instalar o Apache e atualizar o firewall. Vamos executar o primeiro comando, a atualização do gerenciador de pacotes do nosso sistema Ubuntu:

Figura 1 – Atualização do cache do gerenciador de pacotes

```
ubuntu@ip-172-31-20-184:~$ sudo apt-get update|
```

Fonte: Elaborado Pelo Grupo

Após a atualização, vamos instalar o Apache com o seguinte comando:

Figura 2 – Instalação do Apache

```
ubuntu@ip-172-31-20-184:~$ sudo apt-get install apache2|
```

Fonte: Elaborado Pelo Grupo

Confirme a instalação pressionando “Y”, logo após “ENTER”. Agora precisamos ajustar as configurações do firewall executando os seguintes comandos:

Figura 3 – Instalação do Apache

```
ubuntu@ip-172-31-20-184:~$ sudo ufw app list sudo ufw allow in "Apache"|
```

Fonte: Elaborado Pelo Grupo

Agora, logo após todos os comandos executados, deveremos visualizar com o IP do servidor(exemplo <http://12.345.678.91/>) no navegador de sua preferência, você deve encontrar a página padrão do Apache:

Figura 4 – Página do Apache



The screenshot shows the Apache2 Default Page. At the top left is the Ubuntu logo. To its right, the text "Apache2 Default Page" is displayed. Below this, the word "Ubuntu" is written in a large, lowercase, sans-serif font. To the right of "Ubuntu" is a red rectangular button containing the white text "It works!". Below the main title and logo, there is a paragraph of text explaining the purpose of the page. Underneath this text is a section titled "Configuration Overview" with a detailed description of the Apache2 configuration layout.

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   '-- ports.conf
|-- mods-enabled
|   '-- *.load
|   '-- *.conf
|-- conf-enabled
|   '-- *.conf
|-- sites-enabled
|   '-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective `*-available/` counterparts. *These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite` and `a2dissite`.*

Fonte: Elaborado Pelo Grupo

Agora, com tudo configurado, vamos entrar nas pastas “`var/www/html`” para poder criar uma nova pasta e um novo arquivo dentro da mesma, com isso devemos entrar na pasta mencionada.

Figura 5 – comando para entrar na pasta var

```
ubuntu@ip-172-31-20-184:$ cd /var/www/html|
```

Fonte: Elaborado Pelo Grupo

Vamos criar uma pasta com o nome de sua escolha:

Figura 6 – comando para criar pasta

```
ubuntu@ip-172-31-20-184:/var/www/html$ sudo mkdir meuSite|
```

Fonte: Elaborado Pelo Grupo

Com a pasta criada agora vamos entrar na pasta com o comando “**cd (nome da sua pasta criada)**” e vamos criar um arquivo .html para realizar a criação simples de um site e visualizá-lo na nosso ip do servidor:

Figura 7 – comando para criar um arquivo

```
ubuntu@ip-172-31-20-184:/var/www/html$ sudo touch index.html|
```

Fonte: Elaborado Pelo Grupo

Agora com o arquivo criado vamos editá-lo, para conseguir visualizá-lo com o ip do servidor. Antes temos que realizar esse comando para entrar no arquivo .html e configurar com seu código de sua escolha.

Figura 8 – comando para entrar na pasta e comando para editar seu arquivo .html

```
ubuntu@ip-172-31-20-184:/var/www/html$ cd meuSite  
ubuntu@ip-172-31-20-184:/var/www/html/meuSite$ sudo nano index.html|
```

Fonte: Elaborado Pelo Grupo

Com tudo configurado, acesse seu ip no navegador e acesse a pasta escolhida e você irá conseguir visualizar seu site.

Figura 9 – Site configurado



Fonte: Elaborado Pelo Grupo

1.1.2 Protocolo FTP

O protocolo FTP(File Transfer Protocol), traduzido para português como Protocolo de Transferência de Arquivo, permite a transferência de arquivos em um modelo cliente-servidor através de uma conexão padrão web padrão utilizando TCP nas portas 20 a 21. O princípio do funcionamento do FTP ocorre por dois canais de comunicação: o canal de comando e o canal de dados. O canal de comandos é responsável por lidar com as instruções de execução e o que retornam, enquanto o canal de dados se responsabiliza pelo transporte dos arquivos enviados e/ou recebidos. Geralmente a porta 21 é

utilizada para o canal de comando e a 20 para dados. Para iniciar a configuração do FTP, primeiramente é necessário instalar o pacote “**vsftpd**” no servidor remoto.

Figura 10 – Instalação do Pacote vsftpd no servidor remoto

```
tu@ip-172-31-48-141:~$ sudo apt install vsftpd
```

Fonte: Elaborado Pelo Grupo

Após a instalação do pacote é necessário executar os comandos “**sudo service vsftpd restart**” e posteriormente “**sudo service vsftpd status**” para se certificar de que o serviço esteja rodando corretamente.

Figura 11 – Início e Checagem de status do serviço vsftpd

```
tu@ip-172-31-48-141:~$ sudo service vsftpd restart
tu@ip-172-31-48-141:~$ sudo service vsftpd status
vsftpd.service - vsftpd FTP server
   Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; enabled; preset: enabled)
     Active: active (running) since Sat 2025-10-18 00:33:50 UTC; 4s ago
       Process: 1417 ExecStartPre=/bin/mkdir -p /var/run/vsftpd/empty (code=exited, status=0/SUCCESS)
      Main PID: 1420 (vsftpd)
         Tasks: 1 (limit: 1121)
        Memory: 804.0K (peak: 1.1M)
          CPU: 7ms
         CGroup: /system.slice/vsftpd.service
                   └─1420 /usr/sbin/vsftpd /etc/vsftpd.conf

18 00:33:50 ip-172-31-48-141 systemd[1]: Starting vsftpd.service - vsftpd FTP server...
18 00:33:50 ip-172-31-48-141 systemd[1]: Started vsftpd.service - vsftpd FTP server.
```

Fonte: Elaborado Pelo Grupo

Uma conexão pelo protocolo FTP geralmente está associada à porta 20 ou 21. Entretanto esta conexão é insegura e não criptografada, ou seja, os dados de usuário e senha da máquina cliente são expostos em texto plano e as informações tornam-se visíveis. Por isso é importante optar pelo uso do protocolo SFTP(Secure File Transfer Protocol), que possui as mesmas funcionalidades do FTP mas com conexões seguras e criptografadas, assegurando-se de que esses dados sejam disponibilizados no formato de um código humanamente ilegível.

O próximo passo é configurar o firewall no terminal e no console da AWS. Primeiramente será efetuada a configuração no terminal: Para ter certeza de que o firewall está habilitado é necessário executar o comando “**sudo ufw status**”.

Figura 12 – Checagem do status do firewall no terminal com o firewall inativo

```
ip-172-31-48-141:~$ sudo ufw status
inactive
```

Fonte: Elaborado Pelo Grupo

Geralmente, o firewall pelo terminal vai estar desabilitado. Portanto, para habilitá-lo rodar o comando “**sudo ufw enable**”.

Figura 13 – Ativação do firewall no terminal

```
ubuntu@ip-172-31-48-141:~$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
```

Fonte: Elaborado Pelo Grupo

Uma vez habilitado, é possível rodar novamente “**sudo ufw status**” para verificar as configurações de tráfego adicionadas.

Figura 14 – Checagem do status do firewall no terminal

```
ubuntu@ip-172-31-48-141:~$ sudo ufw status
Status: active

To                         Action      From
--                         --          --
20:21/tcp                  ALLOW       Anywhere
12000:12100/tcp             ALLOW       Anywhere
22/tcp                     ALLOW       Anywhere
Apache
80/tcp                     ALLOW       Anywhere
443
990/tcp                    ALLOW       Anywhere
3000:31000/tcp              ALLOW       Anywhere
30000:31000/tcp             ALLOW       Anywhere
5432/tcp                   ALLOW       Anywhere
20:21/tcp (v6)              ALLOW       Anywhere (v6)
12000:12100/tcp (v6)        ALLOW       Anywhere (v6)
22/tcp (v6)                 ALLOW       Anywhere (v6)
Apache (v6)
80/tcp (v6)                 ALLOW       Anywhere (v6)
443 (v6)
990/tcp (v6)                ALLOW       Anywhere (v6)
3000:31000/tcp (v6)         ALLOW       Anywhere (v6)
5432/tcp (v6)               ALLOW       Anywhere (v6)
```

Fonte: Elaborado Pelo Grupo

Se não houver nenhuma configuração anterior de controle de tráfego, a lista geralmente estará vazia. Para o gerenciamento do controle de tráfego dos protocolos é possível utilizar o nome do protocolo ou a porta que utiliza conjuntamente ao(s) protocolo(s) de comunicação. Para permitir o tráfego FTP adicionar o intervalo de portas 20 a 21, portanto rodando o comando “**sudo ufw allow 20:21/tcp**”, uma vez que para o caso apenas o protocolo de comunicação TCP basta.

Figura 15 – Permissão do tráfego para conexões FTP no terminal

```
ip-172-31-48-141:~$ sudo ufw allow 20:21/tcp
| adding existing rule
| adding existing rule (v6)
```

Fonte: Elaborado Pelo Grupo

Como no caso o tráfego já existe, o terminal retornou um output informando que a regra de tráfego já foi adicionada anteriormente.

Também será necessário adicionar o intervalo de portas 12000 a 12100 através do protocolo TCP. Esse intervalo serve para estabelecer as portas de conexão na parte da máquina cliente, visto que no modo ativo portas aleatórias serão escolhidas para a conexão da máquina cliente em uma porta local antes de se conectar à porta do servidor. Com isso, será rodado o comando “**sudo ufw allow 12000:12100/tcp**”.

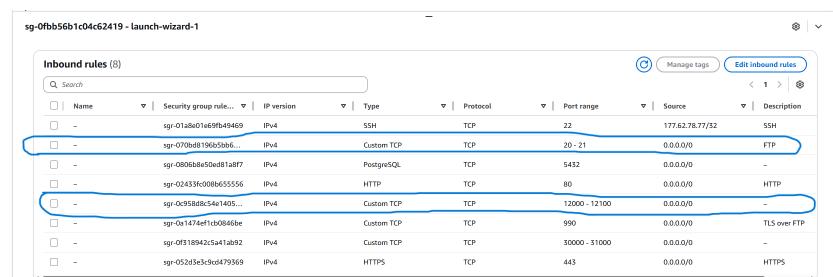
Figura 16 – Permissão do tráfego para conexões FTP no modo passivo no terminal

```
Skiping adding existing rule
Skiping adding existing rule (v6)
ubuntu@ip-172-31-48-141:~$
```

Fonte: Elaborado Pelo Grupo

Uma vez com o firewall configurado no terminal, o mesmo procedimento precisará ser feito no grupo de segurança da instância EC2 pelo console da AWS.

Figura 17 – Permissão do tráfego para FTP na AWS



Fonte: Elaborado Pelo Grupo

Após isso será necessário adicionar as informações do usuário que poderá efetuar as conexões FTP com o servidor. Para isso rodar “**sudo useradd [nome]**” e posteriormente “**sudo passwd [nome]**”, substituindo o campo “[nome]” pelo nome escolhido. No caso o usuário escolhido será “**random**” então os comandos “**sudo useradd random**” e “**sudo passwd random**” serão rodados. Ao rodar “**sudo passwd**” para o respectivo usuário inserir uma senha de escolha e confirmar

Figura 18 – Configuração do usuário que irá utilizar conexões FTP com o servidor

```
ubuntu@ip-172-31-48-141:~$ sudo useradd matheus
ubuntu@ip-172-31-48-141:~$ sudo passwd matheus
New password:
Retype new password:
passwd: password updated successfully
```

Fonte: Elaborado Pelo Grupo

Como o arquivo de configuração gerado pela instalação do vsftpd terá que ser editado, para evitar perdas é importante criar uma configuração de backup, com os dados iniciais. Para isso, será necessário rodar “**sudo cp /etc/vsftpd.conf /etc/vsftpd.conf.bkp**”, que copiará o arquivo original para uma réplica original.

Figura 19 – Cópia do arquivo de configuração vsftpd.conf

```
buntu@ip-172-31-48-141:~$ sudo cp /etc/vsftpd.conf /etc/vsftpd.conf.bkp
buntu@ip-172-31-48-141:~$
```

Fonte: Elaborado Pelo Grupo

Para editar o arquivo vsftpd.conf será necessário utilizar um editor de, como vim ou nano. No caso será utilizado nano, portanto rodar “**sudo nano /etc/vsftpd.conf**” para abrir o editor de texto usando GNU nano.

Figura 20 – Comando para edição do arquivo de configuração vsftpd

```
ubuntu@ip-172-31-48-141:~$ sudo nano /etc/vsftpd.conf
```

Fonte: Elaborado Pelo Grupo

Figura 21 – Tela do arquivo de configuração vsftpd

```
Example config file /etc/vsftpd.conf
The default compiled in settings are fairly paranoid. This sample file
loosens things up a bit, to make the ftp daemon more usable.
Please see vsftpd.conf(5) for all compiled in defaults.

READ THIS: This example file is NOT an exhaustive list of vsftpd options.
Please read the vsftpd.conf(5) manual page to get a full idea of vsftpd's
capabilities.

# Run standalone?  vsftpd can run either from an inetd or as a standalone
# daemon started from an initscript.
#listen=NO

# This directive enables listening on IPv6 sockets. By default, listening
# on the "any host" address (.) will accept connections from both IPv4
# and IPv6 clients. It is not necessary to listen on *both* IPv4 and IPv6
# sockets. If you want that (perhaps because you want to listen on specific
# addresses) then you must run two copies of vsftpd with two configuration
# files.
listen_ipv6=YES

# ALLOW anonymous FTP? (Disabled by default).
anonymous_enable=NO

#userlist_deny=NO
#userlist_file=/etc/vsftpd/user_list
#tcp_wrappers=NO
#PAM de Compartilhamento
local_root=/home/random02/dados
anon_root=/var/ftp
allow_writeable_chroot=YES
#Uncomment this to allow local users to log in.
local_enable=YES
#Uncomment this to enable any form of FTP write command.
write_enable=YES

# Default umask for local users is 077. You may wish to change this to 022,
# if you expect that (022 is used by most other Ftp'd's)
local_umask=022

# Uncomment this to allow the anonymous FTP user to upload files. This only
# has an effect if the above global write enable is activated. Also, you will
```

Fonte: Elaborado Pelo Grupo

No caso o arquivo já está com as configurações corretas, mas para adicioná-las precisará inserir o seguinte trecho no arquivo: “**userlist_deny=NO** **userlist_file=/etc/vsftpd/user_list** **tcp_wrappers=NO** **write_enable=YES**

local_root=/home/[nome]/dados chroot_local_user=YES allow_writeable_chroot=YES”.

Substitua “[nome]” pelo nome do usuário escolhido, no caso “**/home/random/dados**”.

Estas configurações permitem gerenciar corretamente os usuários adicionados à lista de usuários para acessar o vsftpd, fornecem permissão de escrita e estabelecem o caminho da pasta de dados para ser gerenciada pelo dado usuário, no caso dentro do diretório /home da máquina virtual.

Será necessário executar o comando “**sudo mkdir -p /etc/vsftpd**” para criar a pasta e

posteriormente "tp" para adicionar o arquivo de configuração dos usuários permitidos e o inserir o nome do usuário escolhido, como especificado no caminho fornecido ao arquivo vsftpd.conf. A seguir executar “**sudo mkdir -p /home/[nome]/dados**” e “**sudo chown [nome] /home/[nome]/dados**” para que o usuário determinado em “[nome]” tenha acesso a escrita e leitura em “/home/[nome]/dados”, substituindo “[nome]” pelo nome do usuário. No caso, os comandos “**sudo mkdir -p /home/random/dados**” e “**sudo chown random /home/random/dados**”.

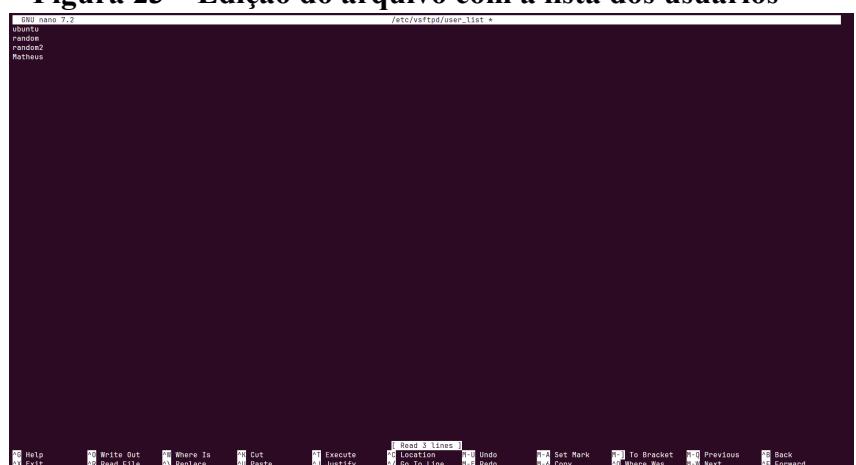
Figura 22 – Comandos para criar o caminho para o arquivo de lista de usuários



```
51-48-141:~$ sudo mkdir -p /etc/vsftpd
51-48-141:~$ sudo nano /etc/vsftpd/user_list
51-48-141:~$
```

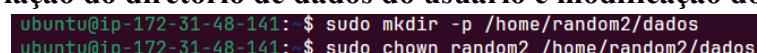
Fonte: Elaborado Pelo Grupo

Figura 23 – Edição do arquivo com a lista dos usuários



Fonte: Elaborado Pelo Grupo

Figura 24 – Criação do diretório de dados do usuário e modificação do dono do diretório



```
ubuntu@ip-172-31-48-141:~$ sudo mkdir -p /home/random2/dados
ubuntu@ip-172-31-48-141:~$ sudo chown random2 /home/random2/dados
```

Fonte: Elaborado Pelo Grupo

Com todas as configurações finalizadas, será necessário reiniciar o serviço “**vsftpd**”. Portanto rodar “**sudo service vsftpd restart**” para se certificar de que todas as alterações

sejam aplicadas.

Figura 25 – Comando para reiniciar o serviço vsftpd

```
ubuntu@ip-172-31-48-141: ~ $ sudo service vsftpd restart  
ubuntu@ip-172-31-48-141: ~ $
```

Fonte: Elaborado Pelo Grupo

Agora será possível utilizar o comando “**sudo ftp [nome]@localhost**”, onde “[nome]” substitui o nome do usuário escolhido, no caso ficaria “**sudo ftp random@localhost**”, o que

irá abrir a conexão ftp via terminal. Também iremos executar “**mkdir teste**” para criar uma pasta chamada Teste pela CLI do vsftpd.

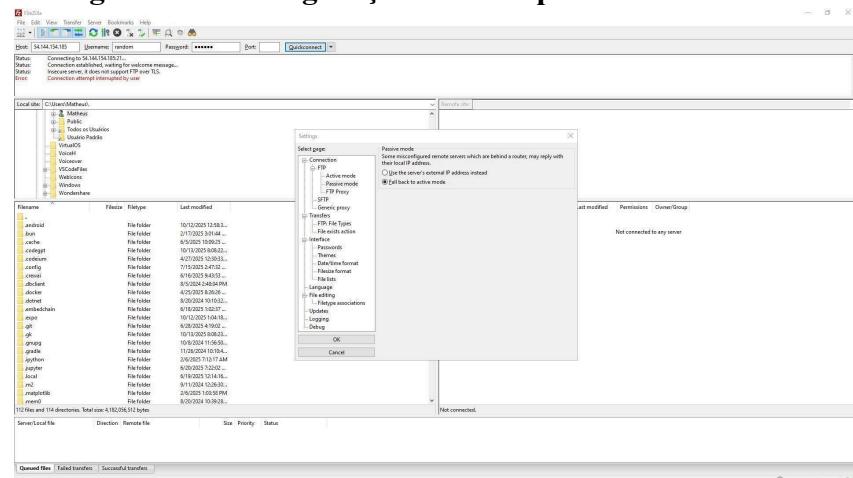
Figura 26 – Inicialização e uso do FTP pelo terminal

```
Connected to localhost.  
220 (vsFTPd 3.0.5)  
331 Please specify the password.  
Password:  
230 Login successful.  
Remote system type is UNIX.  
Using binary mode to transfer files.  
ftp> mkdir teste3  
257 "/teste3" created  
ftp>
```

Fonte: Elaborado Pelo Grupo

Outra maneira de efetuar a conexão FTP é utilizando o Filezilla. Basta instalar o Filezilla Client e executar na máquina. Após isso, será necessário configurá-lo para sempre optar por conexões no modo ativo por conta de configurações de firewall. Para isso vá em “**Edit**”, depois em “**Settings**” e “**Passive Mode**” e então escolha “**Fall back to active mode**”.

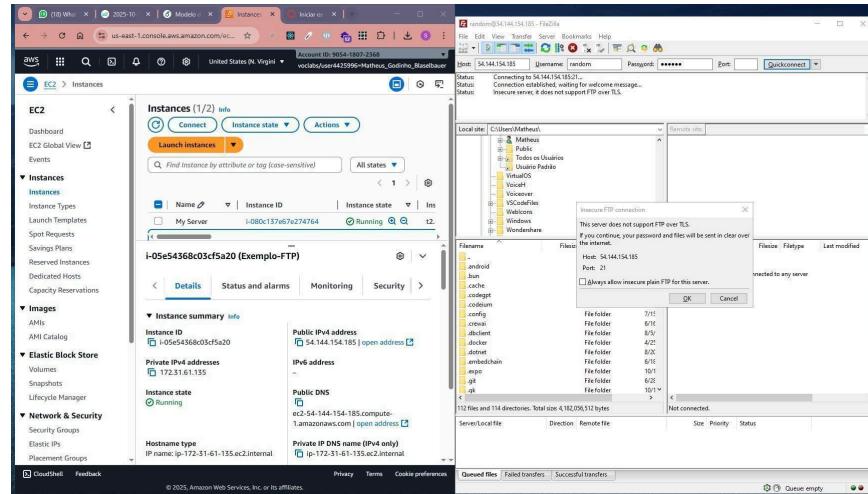
Figura 27 – Configuração Filezilla para modo ativo



Fonte: Elaborado Pelo Grupo

Após isso, inserir as informações do IP público do servidor da AWS e os dados do usuário para se conectar ao FTP. Aparecerá um aviso para que permita a conexão FTP insegura, e será necessário clicar em “OK” para abrir a conexão.

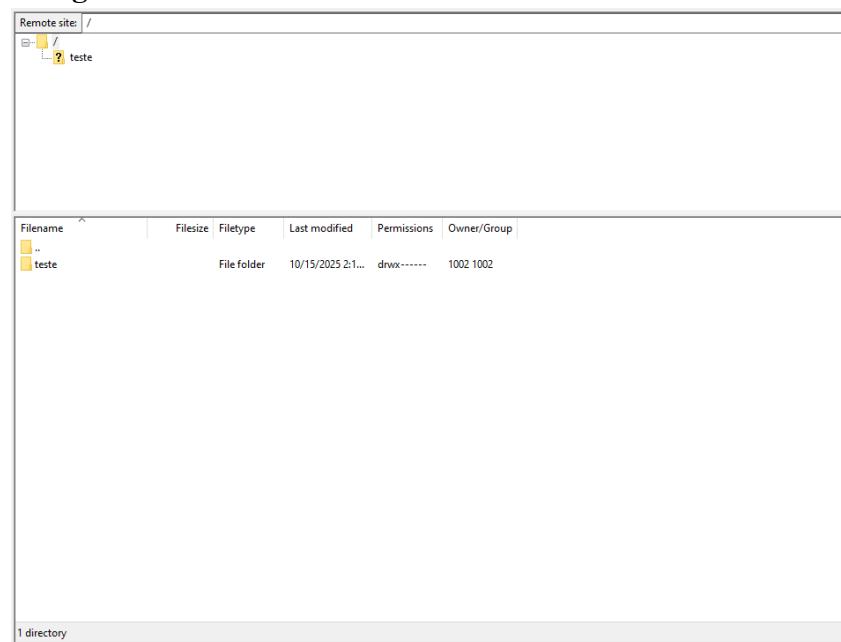
Figura 28 – Conexão FTP Filezilla com o servidor AWS



Fonte: Elaborado Pelo Grupo

Será possível ver que há uma pasta chamada “Teste”, criada anteriormente pela CLI do ftp no terminal.

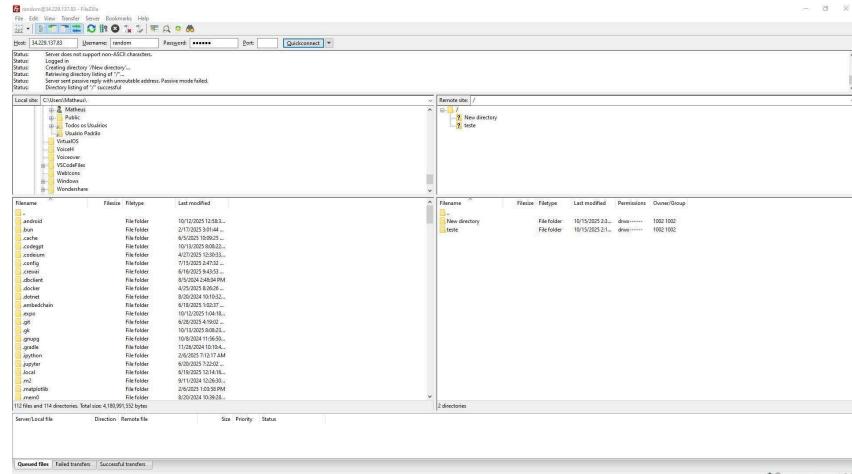
Figura 29 – Conexão FTP Filezilla com o servidor AWS 2



Fonte: Elaborado Pelo Grupo

Pode-se criar uma pasta ou arquivo direto pelo Filezilla, assim como no terminal, sendo necessário clicar com o botão direito no diretório e clicar em criar, com o nome desejado.

Figura 30 – Conexão FTP Filezilla com o servidor AWS 3



Fonte: Elaborado Pelo Grupo

1.1.3 Banco de Dados Local PostgreSQL

A escolha para a configuração do banco de dados foi utilizar o PostgreSQL dentro de uma instância EC2, que é o modo de configuração local. Para inicializar a configuração, é necessário instalar o postgresql na máquina virtual. Para isso, no terminal rode "sudo apt install postgresql postgresql-contrib".

Figura 31 – Instalação do PostgreSQL na máquina EC2

Անձնագիր-172-31-48-141. Տօնությունը կազմված է PostgreSQL-ում:

Fonte: Elaborado Pelo Grupo

Com isso os arquivos e executáveis necessários serão instalados. Feito isso, um usuário “**postgres**” será instalado na instância. Com isso, ao executar o comando “**sudo -i -u postgres**” terá- se o acesso a este usuário.

Figura 32 – Acesso ao usuário postgres

```
ubuntu@ip-172-31-48-141:~$ sudo -i -u postgres  
postgres@ip-172-31-48-141:~$
```

Fonte: Elaborado Pelo Grupo

Uma vez no terminal, será criado um banco de dados com o nome a sua escolha através do comando “**createdb [nome]**”, onde “[**nome**]” será substituído pelo nome do banco de dados que deseja criar. No caso, como já existe um banco de dados chamado de teste no sistema, o terminal retornou um erro informando esta ocorrência e teve que ser criado um outro banco de dados chamado “**teste2**”.

Figura 33 – Criação do banco de dados

```
s@ip-172-31-48-141:~$ createdb meu_db  
s@ip-172-31-48-141:~$
```

Fonte: Elaborado Pelo Grupo

Uma vez com o banco de dados criado, haverá a necessidade de se acessar o executável do banco de dados para gerar comandos SQL de acordo com o dialeto PostgreSQL. Para isso basta iniciar o cliente psql. É muito importante não se esquecer de inserir ";" ao final de cada query, caso contrário os resultados esperados não serão executados.

Figura 34 – Acesso aos comandos SQL com cliente psql

```
postgres@ip-172-31-48-141:~$ createdb meu_db  
postgres@ip-172-31-48-141:~$ psql  
psql (16.10 (Ubuntu 16.10-0ubuntu0.24.04.1))  
Type "help" for help.  
postgres=#
```

Fonte: Elaborado Pelo Grupo

Será criado um usuário através do nome da ROLE escolhida e senha. Para isso, executar o comando SQL “**CREATE USER [role] WITH ENCRYPTED PASSWORD '[pass]';**”, trocando “**role**” pelo nome do usuário e “**pass**” pela senha desejada ao ser requisitado o acesso ao banco por meio deste usuário.

Figura 35 – Criação de credenciais de acesso ao banco de dados com permissionamento

```
postgres=# CREATE USER qualquer WITH ENCRYPTED PASSWORD 'qualquer';
CREATE ROLE
postgres=#

```

Fonte: Elaborado Pelo Grupo

Para fornecer as permissões para a ROLE criada, executar “**GRANT ALL PRIVILEGES ON DATABASE [banco] TO [role]**”, onde “**banco**” é o nome da base de dados e “**role**” corresponde ao nome da role usada para atribuir ao usuário.

Figura 36 – Permissionamento ao cargo criado

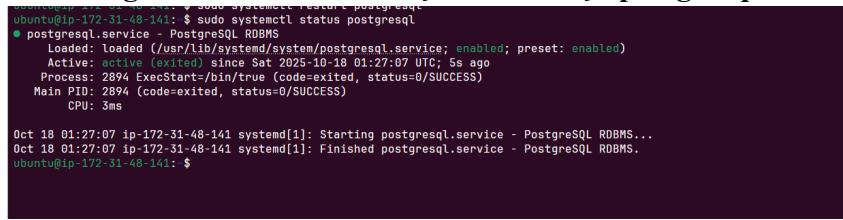
```
postgres=# GRANT ALL PRIVILEGES ON DATABASE meu_db to qualquer;
GRANT
postgres=#

```

Fonte: Elaborado Pelo Grupo

Feito isso, pode-se fazer o logout do usuário postgres e voltar ao usuário ubuntu da instância EC2 para reinicializar o serviço do banco de dados e ter certeza de que todas as configurações foram devidamente aplicadas.

Figura 37 – Reinicialização do serviço postgresql



```
ubuntu@ip-172-31-48-141: ~ $ sudo systemctl restart postgresql
● postgresql.service - PostgreSQL RDBMS
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
   Active: active (exited) since Sat 2025-10-18 01:27:07 UTC; 5s ago
     Process: 2894 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
   Main PID: 2894 (code=exited, status=0/SUCCESS)
      CPU: 3ms

Oct 18 01:27:07 ip-172-31-48-141 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Oct 18 01:27:07 ip-172-31-48-141 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
ubuntu@ip-172-31-48-141: ~ $
```

Fonte: Elaborado Pelo Grupo

Com o serviço reinicializado, será necessário configurar os arquivos gerados pelo pacote postgresql. Para isso, precisaremos editar dois arquivos principais de configuração gerados no caminho “**/etc/postgresql/[versão]/main**”, onde “[versão]” substitui a versão do postgres instalada na máquina

Figura 38– Caminho para editar os arquivos

```
ip-172-31-48-141: $ cd /etc/postgresql/16/main
ip-172-31-48-141:/etc/postgresql/16/main$ ls
environment pg_ctl.conf pg_hba.conf pg_ident.conf postgresql.conf start.conf
ip-172-31-48-141:/etc/postgresql/16/main$
```

Fonte: Elaborado Pelo Grupo

Execute o editor de texto Linux baseado em preferência. No caso será utilizado o GNU nano para modificar inicialmente o conteúdo do arquivo “**postgresql.conf**”. Portanto será rodado “**sudo nano postgresql.conf**”

Figura 39 – Comando para editar o arquivo com GNU nano

```
ubuntu@ip-172-31-48-141:/etc/postgresql/16/main$ sudo nano postgresql.conf
```

Fonte: Elaborado Pelo Grupo

Este arquivo é responsável por gerenciar os caminhos e para quais computadores/IPs o banco de dados ficará disposto à conexão. Por isso, será necessário localizar o campo “**listen_addresses**” e o valor mapeado, trocando para “*” como forma de aceitar conexões com qualquer outro host. Certificar-se inclusive de que o caminho para “**hba_file**” consta como o correto. A porta fica a critério de escolha, mas geralmente se utiliza o padrão(5432).

Figura 40 – Configuração do arquivo postgresql.conf

```
data_directory = '/var/lib/postgresql/16/main'          # use data in another directory
                                                        # (change requires restart)
hba_file = '/etc/postgresql/16/main/pg_hba.conf'       # host-based authentication file
                                                        # (change requires restart)
ident_file = '/etc/postgresql/16/main/pg_ident.conf'   # ident configuration file
                                                        # (change requires restart)

# If external_pid_file is not explicitly set, no extra PID file is written.
external_pid_file = '/var/run/postgresql/16-main.pid'    # write an extra PID file
                                                        # (change requires restart)

#-----
# CONNECTIONS AND AUTHENTICATION
#-----

# - Connection Settings -

listen_addresses = '*'  # what IP address(es) to listen on;
                        # comma-separated list of addresses;
                        # defaults to 'localhost'; use '*' for all
                        # (change requires restart)
port = 5432             # (change requires restart)
max_connections = 100    # (change requires restart)
#reserved_connections = 0 # (change requires restart)
#superuser_reserved_connections = 3 # (change requires restart)
```

Fonte: Elaborado Pelo Grupo

Em seguida, será necessário editar o arquivo “**pg_hba.conf**”, referenciado anteriormente no arquivo “**postgresql.conf**”. O “**pg_hba.conf**” é responsável por autenticar as conexões que são efetuadas através dos IPs das máquinas que requerem o acesso ao banco de dados.

Rodar “**sudo nano pg_hba.conf**”.

Figura 41 – Comando para editar o arquivo com GNU nano

```
ubuntu@ip-172-31-48-141:/etc/postgresql/16/main$ sudo nano pg_hba.conf
```

Fonte: Elaborado Pelo Grupo

Ao ser exibido, o arquivo demonstrará várias configurações, entretanto é no final do arquivo que se encontram as configurações de autenticação de hosts

Dentre essas configurações será necessário adicionar uma linha setada como “**host**”, utilizando “**all**” no IP “**0.0.0/0**”, ou seja, qualquer IP, e no modo “**md5**”.

Para aplicar as alterações, reinicialize novamente o serviço postgresql.

Figura 42 – Comando para editar o arquivo com GNU nano

```
# DO NOT DISABLE!
# If you change this first entry you will need to make sure that the
# database superuser can access the database using some other method.
# Noninteractive access to all databases is required during automatic
# maintenance (custom daily cronjobs, replication, and similar tasks).
#
# Database administrative login by Unix domain socket
local    all            postgres                      peer
#
# TYPE  DATABASE        USER        ADDRESS             METHOD
#
# "local" is for Unix domain socket connections only
local    all            all                            peer
# IPv4 local connections:
host     all            all      127.0.0.1/32       scram-sha-256
# IPv6 local connections:
host     all            all      ::1/128           scram-sha-256
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication   all                            peer
host   replication   all      127.0.0.1/32       scram-sha-256
host   replication   all      ::1/128           scram-sha-256
host   all            all      0.0.0.0/0          md5
```

Fonte: Elaborado Pelo Grupo

Figura 43 – Reinicialização do serviço postgresql

```
1:~$ sudo systemctl restart postgresql
1:~$ sudo systemctl status postgresql
   PostgreSQL RDBMS
   [systemd/system/postgresql.service; enabled; preset: enabled]
   ● postgresql.service - PostgreSQL RDBMS
       Active: active (exited) since Sat 2025-10-18 01:33:54 UTC; 4s ago
         Start: /bin/true (code=exited, status=0/SUCCESS)
        Stop: /bin/true (code=exited, status=0/SUCCESS)

   -31-48-141 systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
   -31-48-141 systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
1:~$
```

Fonte: Elaborado Pelo Grupo

Com isso, o servidor que hospeda o banco de dados já está configurado. Entretanto, para que as máquinas cliente possam se conectar ao banco de dados é preciso expor a porta em que o serviço postgresql irá rodar, especificada anteriormente no arquivo de configuração “**postgresql.conf**”. No caso, como especificado anteriormente, a porta padrão “**5432**” está sendo utilizada para que o servidor aceite as conexões mapeadas para esta porta. Portanto, será necessário configurar isso para o firewall do servidor no console AWS e no terminal. Rodar “**sudo ufw allow [porta]**”, em que “[porta]” será a porta mapeada como dito, no caso “**5432**”.

Figura 44 – Permissão de tráfego para a porta 5432 pelo terminal

```
Oct 18 01:55:54 ip-172-31-48-141 systemd[1]: Finished postgresql.service PostgreSQL RDBMS.  
ubuntu@ip-172-31-48-141:~$ sudo ufw allow 5432/tcp  
Skipping adding existing rule  
Skipping adding existing rule (v6)  
ubuntu@ip-172-31-48-141:~$
```

Fonte: Elaborado Pelo Grupo

Como a regra já havia sido adicionada, houve um aviso. Para a AWS, basta ir no grupo de segurança da instância e adicionar a mesma regra.

Figura 45 – Permissão de tráfego para a porta 5432 no Console AWS



Fonte: Elaborado Pelo Grupo

O teste será efetuado de duas maneiras: pelo terminal e pgAdmin. Para fazer o teste via terminal utilize uma outra instância EC2 na AWS no mesmo grupo de segurança ou em um grupo que tenha permissão para tráfego na porta 5432 e, no terminal, execute “**sudo psql -h “[ip]”-U [role] -d [db]**”. Os campos “[ip]”, “[role]” e “[db]” se referem respectivamente ao IP do servidor: no caso de estar na mesma rede virtual privada criada pela AWS pode utilizar o IP privado do servidor, do contrário utilizar o IP público, ao usuário que deseja utilizar para estabelecer a conexão e por fim em qual banco de dados estabelecer tal conexão.

Figura 46 – Conexão com o banco de dados pelo terminal

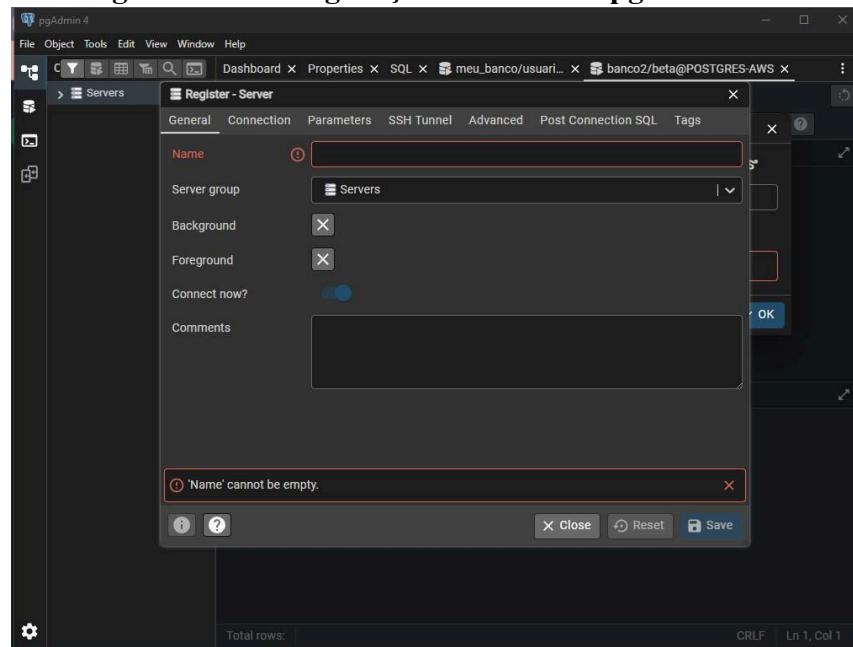
```
ubuntu@ip-172-31-51-29:~$ sudo psql -h "172.31.48.141" -U qualquer -d meu_db
Password for user qualquer:
psql (16.10 (Ubuntu 16.10-0ubuntu0.24.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

meu_db=>
```

Fonte: Elaborado Pelo Grupo

Já pelo pgAdmin, o usuário terá que acessar o menu “Object”, “register” e “Server”, o que irá abrir a janela de configurações.

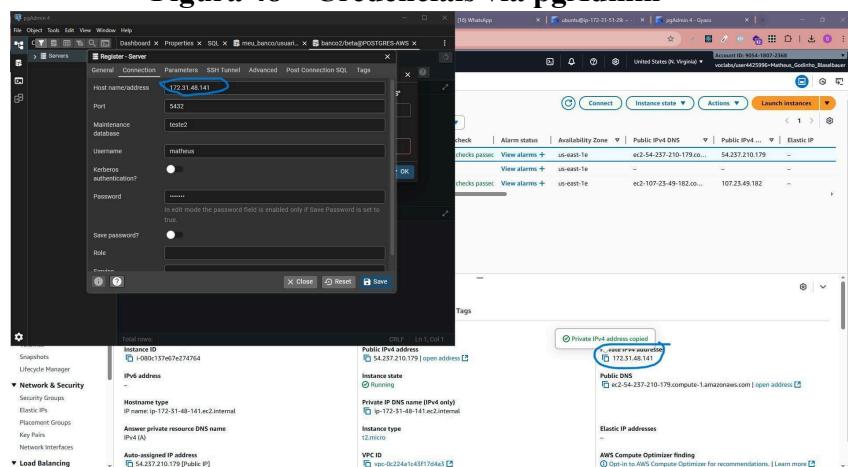
Figura 47 – Configuração conexão via pgAdmin



Fonte: Elaborado Pelo Grupo

Insira um nome qualquer a sua escolha para o grupo de servidores, pois não pode estar vazio. Depois vá à aba “Connection” ou “Conexões”, preencha as informações corretamente e clique em “Save” ou “Salvar”. Pronto, o cliente pgAdmin já está conectado ao banco de dados.

Figura 48 – Credenciais via pgAdmin



Fonte: Elaborado Pelo Grupo

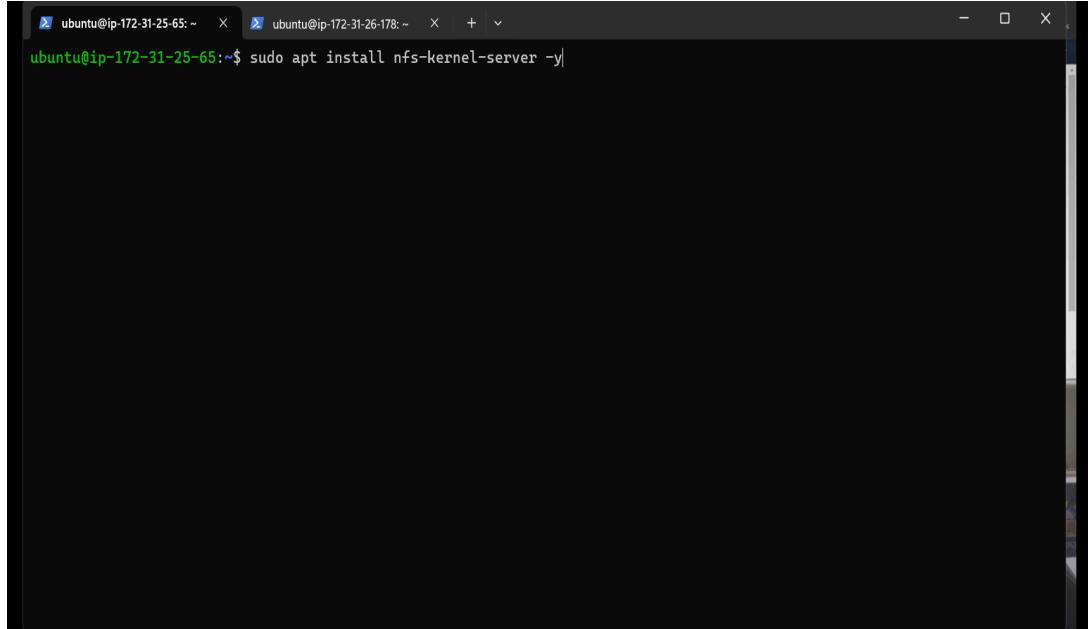
1.1.4 Network File System

O protocolo NFS (Network File System), traduzido como Sistema de Arquivos em Rede, permite que um computador cliente acesse diretórios e arquivos em um servidor pela rede como se eles estivessem em seu próprio disco local. Ele opera em uma arquitetura

cliente-servidor e utiliza o protocolo RPC (Remote Procedure Call) para a comunicação. O funcionamento baseia-se em dois conceitos principais: a exportação no lado do servidor, onde se define qual diretório será compartilhado, e a montagem (mount) no lado do cliente, que anexa esse diretório remoto a uma pasta local.

Para configurar um servidor NFS em um sistema Linux, primeiramente é necessário instalar o pacote “**nfs-kernel-server**” em uma máquina que vamos usar como servidor.

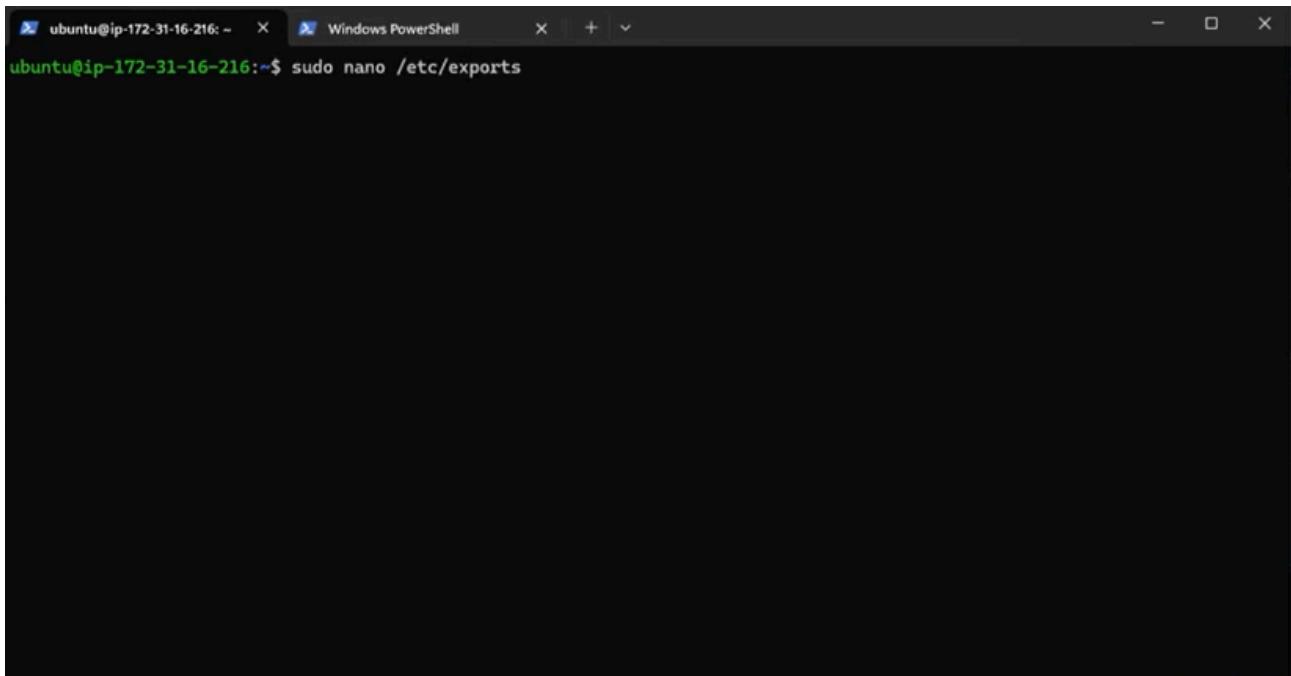
Figura 49– Instalação do nfs-server



Fonte: Elaborado Pelo Grupo

Com isso vamos ter acesso a parte do arquivo de configuração dentro do /etc(exports, então basta acessá-lo usando por exemplo o editor **NANO** como foi feito na imagem abaixo:

Figura 50 – acessando as configurações

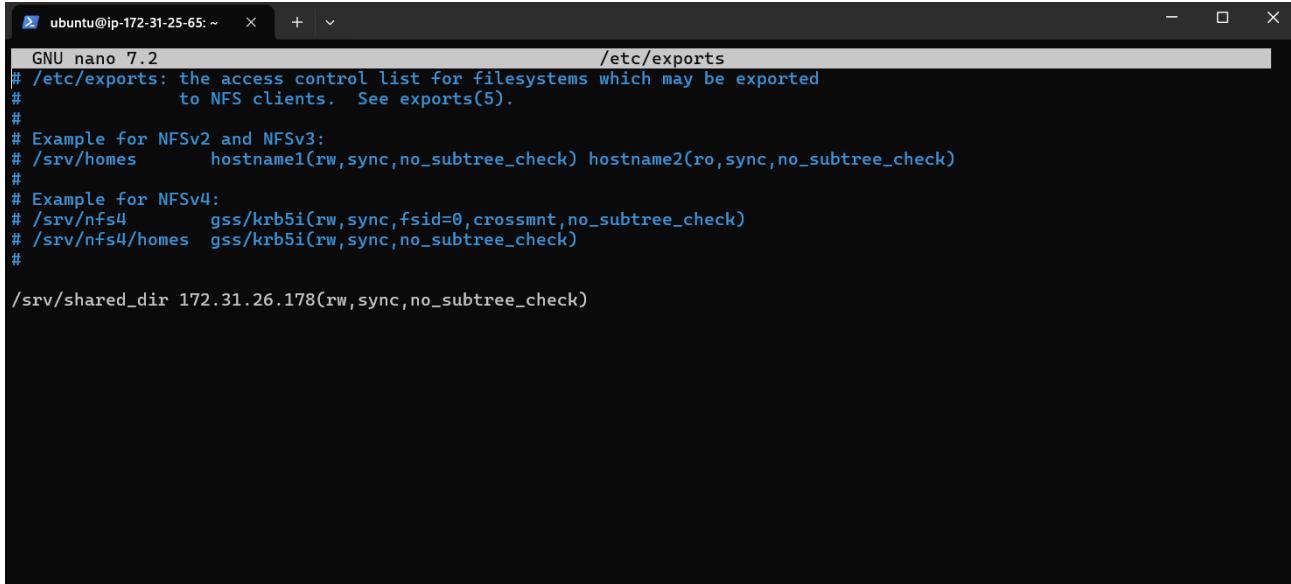


A screenshot of a terminal window titled "Windows PowerShell". The window shows a single line of text: "ubuntu@ip-172-31-16-216:~\$ sudo nano /etc/exports". The background of the terminal is black, and the text is white.

Fonte: Elaborado Pelo Grupo

Já nas configurações a gente vai usar o comando “/srv/shared_dir 172.31.26.178(rw, sync, no_subtree_check)”, o “/srv” é para deixar dentro da pasta srv mesmo, já o “/shared_dir” é a pasta compartilhada que vamos criar, em seguida vem a faixa, é uma boa pratica adicionar os IPs de quem vai acessar esse serviço, e então esse IP vai ter essas propriedades que estão entre parênteses(escrita, sincronização e o “no_subtree_check” que não deixa ficar com o desempenho muito lento).

Figura 51 – adicionando as configurações necessárias

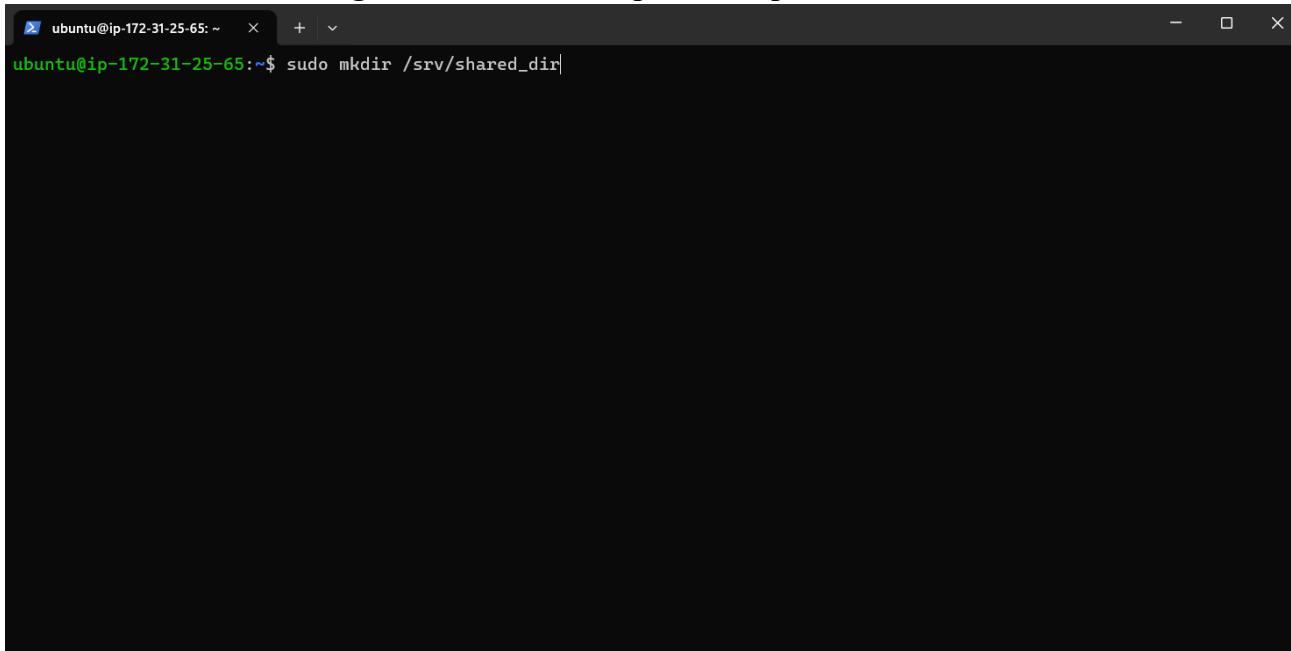


```
GNU nano 7.2 /etc/exports
# /etc/exports: the access control list for filesystems which may be exported
#           to NFS clients. See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes      hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4      gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/srv/shared_dir 172.31.26.178(rw,sync,no_subtree_check)
```

Fonte: Elaborado Pelo Grupo

Logo em seguida, salvamos essa configuração e então criamos a pasta “**shared_dir**” com o comando “**sudo mkdir /srv/shared_dir**”.

Figura 52 – Criando a pasta compartilhada

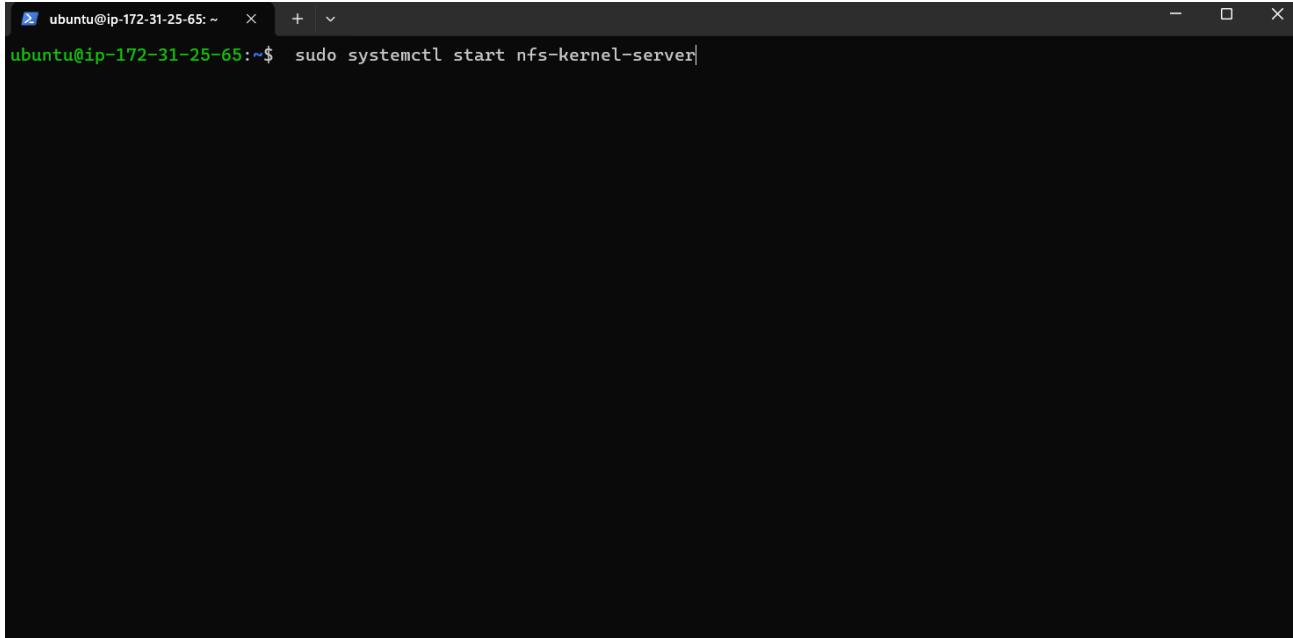


```
ubuntu@ip-172-31-25-65:~$ sudo mkdir /srv/shared_dir
```

Fonte: Elaborado Pelo Grupo

Depois disso vamos iniciar o serviço com o comando “**sudo systemctl start nfs-kernel-server**”.

Figura 53 – Iniciando o serviço

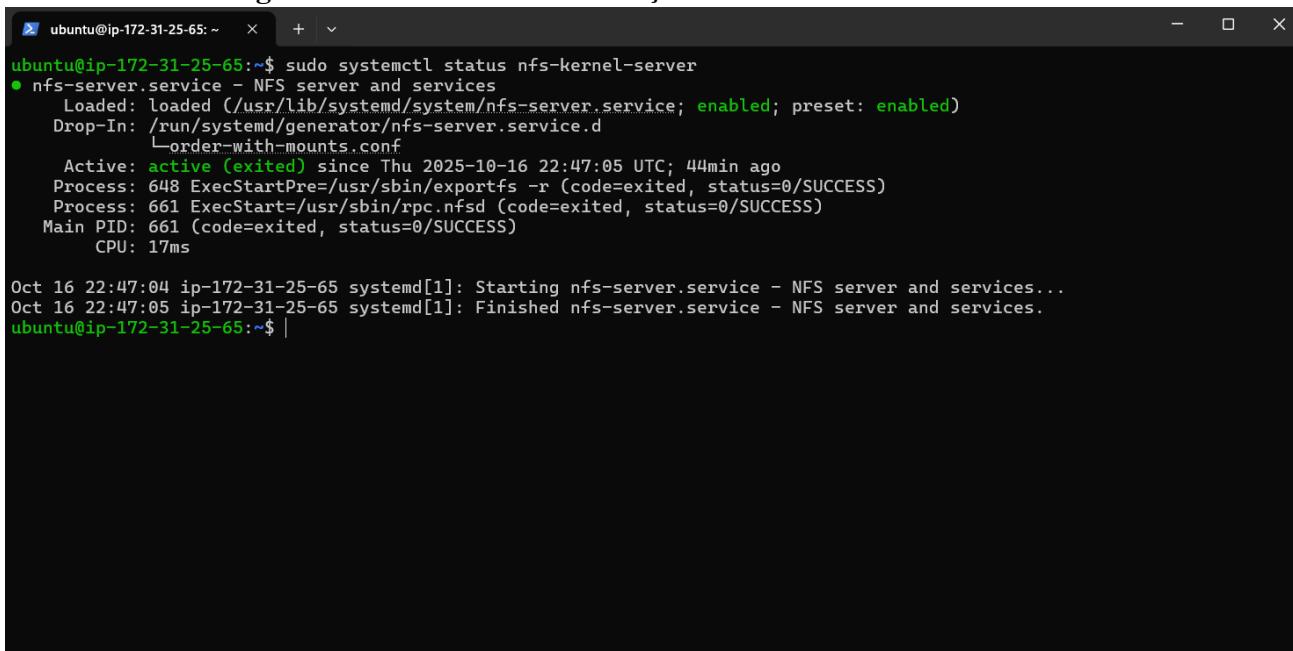


```
ubuntu@ip-172-31-25-65:~$ sudo systemctl start nfs-kernel-server
```

Fonte: Elaborado Pelo Grupo

Para confirmar que foi iniciado com sucesso você pode usar o comando “**sudo systemctl status nfs-kernel-server**”

Figura 54 – Testando se o serviço foi iniciado com sucesso



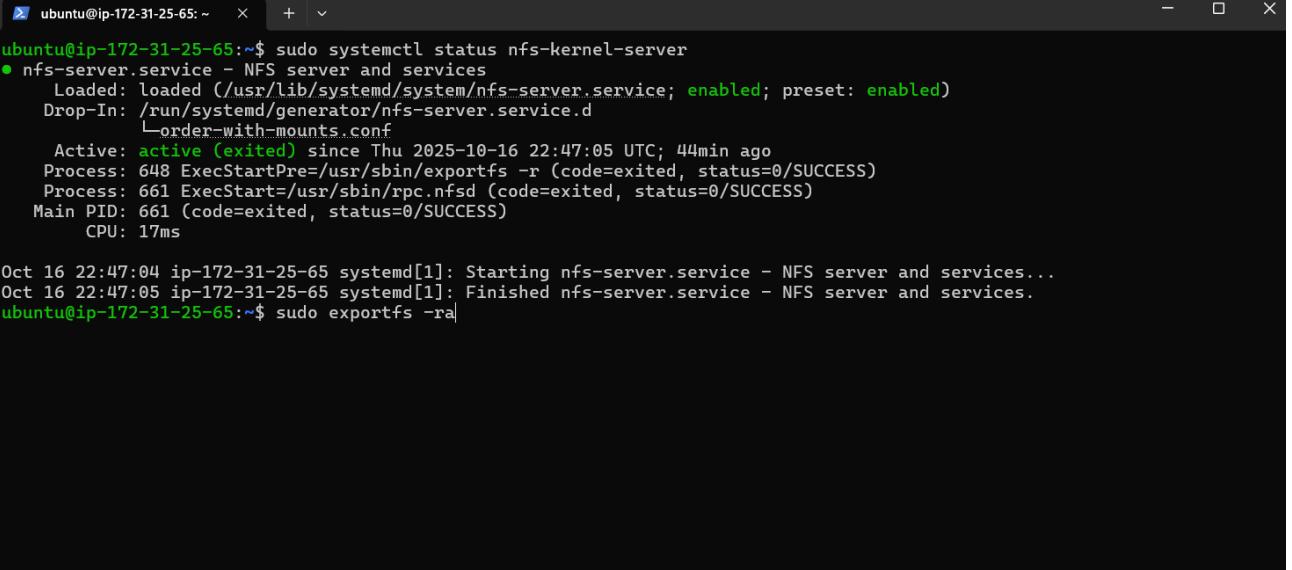
```
ubuntu@ip-172-31-25-65:~$ sudo systemctl status nfs-kernel-server
● nfs-server.service - NFS server and services
  Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; enabled; preset: enabled)
  Drop-In: /run/systemd/generator/nfs-server.service.d
            └─order-with-mounts.conf
    Active: active (exited) since Thu 2025-10-16 22:47:05 UTC; 44min ago
      Process: 648 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUCCESS)
      Process: 661 ExecStart=/usr/sbin/rpc.nfsd (code=exited, status=0/SUCCESS)
        Main PID: 661 (code=exited, status=0/SUCCESS)
         CPU: 17ms

Oct 16 22:47:04 ip-172-31-25-65 systemd[1]: Starting nfs-server.service - NFS server and services...
Oct 16 22:47:05 ip-172-31-25-65 systemd[1]: Finished nfs-server.service - NFS server and services.
ubuntu@ip-172-31-25-65:~$ |
```

Fonte: Elaborado Pelo Grupo

Nota-se que precisa estar tudo verde para dizer que deu certo. Em seguida vamos usar o comando “**sudo exportfs -ra**” para aplicar as configurações que foram feitas anteriormente

Figura 55 – Aplicando as configurações feitas



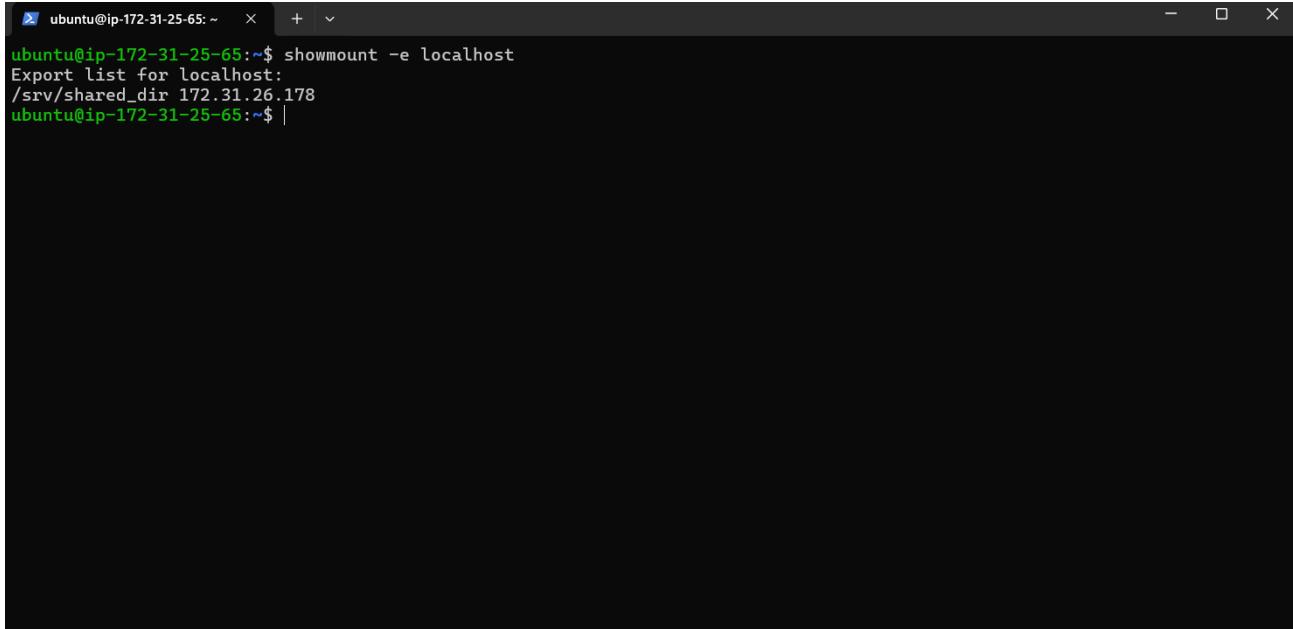
```
ubuntu@ip-172-31-25-65:~$ sudo systemctl status nfs-kernel-server
● nfs-server.service - NFS server and services
  Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; enabled; preset: enabled)
  Drop-In: /run/systemd/generator/nfs-server.service.d
            └─order-with-mounts.conf
    Active: active (exited) since Thu 2025-10-16 22:47:05 UTC; 44min ago
      Process: 648 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUCCESS)
      Process: 661 ExecStart=/usr/sbin/rpc.nfsd (code=exited, status=0/SUCCESS)
    Main PID: 661 (code=exited, status=0/SUCCESS)
       CPU: 17ms

Oct 16 22:47:04 ip-172-31-25-65 systemd[1]: Starting nfs-server.service - NFS server and services...
Oct 16 22:47:05 ip-172-31-25-65 systemd[1]: Finished nfs-server.service - NFS server and services.
ubuntu@ip-172-31-25-65:~$ sudo exportfs -ra
```

Fonte: Elaborado Pelo Grupo

Um teste que pode ser feito para ver o que está sendo oferecido pelo servidor é o do comando **showmount -e localhost** que vai mostrar todos os diretórios que estão sendo exportados nesse servidor, como mostra a imagem abaixo:

Figura 56 – Teste para ver diretórios exportados

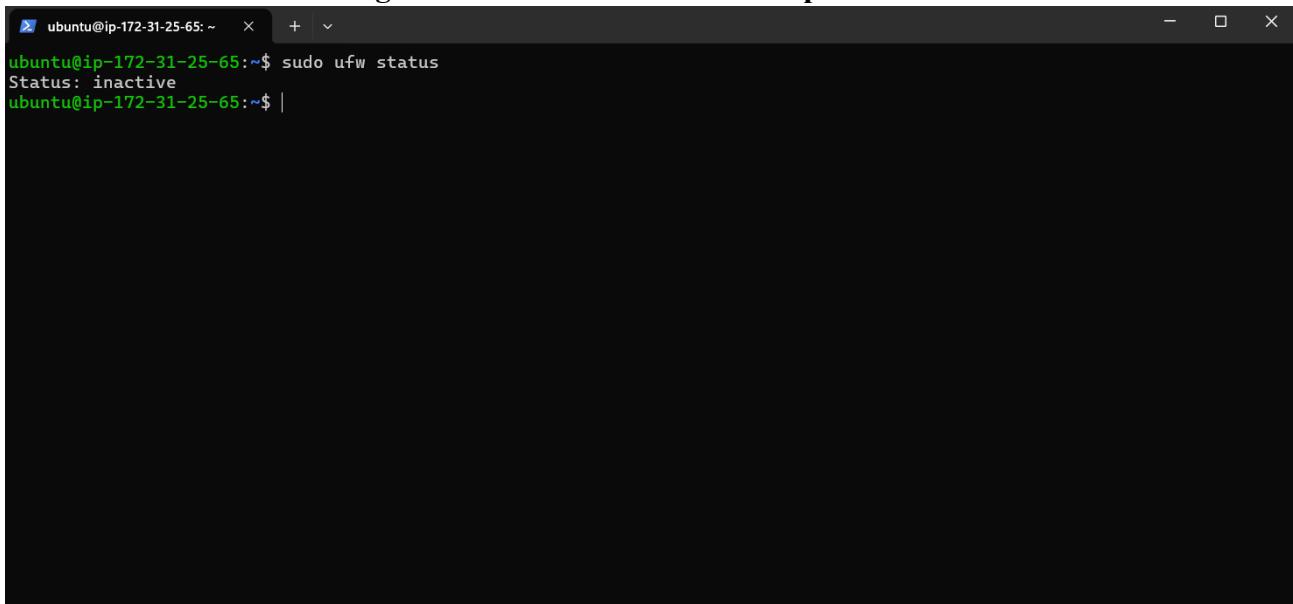


```
ubuntu@ip-172-31-25-65:~$ showmount -e localhost
Export list for localhost:
/srv/shared_dir 172.31.26.178
ubuntu@ip-172-31-25-65:~$ |
```

Fonte: Elaborado Pelo Grupo

Agora precisamos liberar as portas do serviço para saber se está tudo ok, a gente consegue olhar usando o comando **sudo ufw status**.

Figura 57 – Testar os estados das portas



```
ubuntu@ip-172-31-25-65:~$ sudo ufw status
Status: inactive
ubuntu@ip-172-31-25-65:~$ |
```

Fonte: Elaborado Pelo Grupo

Para liberar as portas nós vamos no AWS Management Console, em seguida vamos na instância servidor que no meu caso foi o SrvUbuntu1.

Figura 58 – Acessando a instancia servidor

The screenshot shows the AWS CloudWatch Metrics interface for an Amazon Linux 2 instance. The instance ID is i-0a5fd07de8fcf078d. The instance is currently executing. It has a public IPv4 address of 3.85.44.235 and a private DNS name of ec2-3-85-44-235.compute-1.amazonaws.com. The instance type is t3.micro and it is running in a VPC with ID vpc-0de67ca73ac91e2e5. The subnet is subnet-065ca60ba1de40006. The instance is associated with an IAM role and is part of an Auto Scaling group.

Detalhe	Valor
ID da instância	i-0a5fd07de8fcf078d
Endereço IPv6	-
Tipo de nome do host	Nome do IP: ip-172-31-25-65.ec2.internal
Nome do DNS do recurso privado de resposta	IPv4 (A)
Endereço IP atribuído automaticamente	3.85.44.235 [IP público]
Função do IAM	-
Endereço IPv4 público	3.85.44.235 endereço aberto
Estado da instância	Executando
Nome do DNS de IP privado (somente IPv4)	ip-172-31-25-65.ec2.internal
Tipo de instância	t3.micro
ID da VPC	vpc-0de67ca73ac91e2e5
ID da sub-rede	subnet-065ca60ba1de40006
Endereços IP elásticos	-
Descoberta do AWS Compute Optimizer	Opte por participar do AWS Compute Optimizer para obter recomendações.
Nome do Grupo do Auto Scaling	-

Fonte: Elaborado Pelo Grupo

Depois vamos em segurança-> grupo de segurança-> editar regras de entrada e adicionar duas regra, uma de TCP personalizada com a porta 111 e outra TCP personalizada 2049:

Figura 59 – Adicionando as portas 2049 e 111 nas regras de entrada

The screenshot shows the AWS Security Groups interface for a security group named sgr-0d182b7f3c51b125f. The interface lists four existing ingress rules and allows for adding new ones. The new rules being added are:

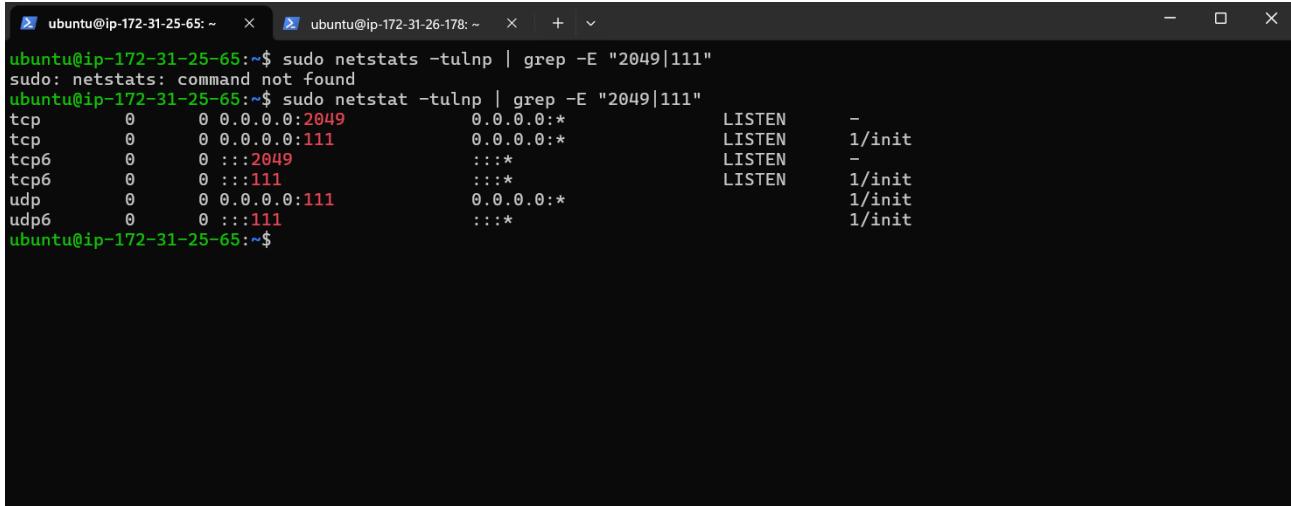
- A rule allowing ICMP traffic from 172.31.26.178/32 to port 111.
- A rule allowing SSH traffic from 0.0.0.0/0 to port 22.
- A rule allowing TCP personalizado traffic from 0.0.0.0/0 to port 111.
- A rule allowing NFS traffic from 0.0.0.0/0 to port 2049.

ID da regra do grupo de segurança	Tipo	Protocolo	Intervalo de portas	Origem	Descrição - opcional
sgr-0d182b7f3c51b125f	Todos os ICMPs - IPv4	ICMP	Todo	Pesso...	Ping ICMP
sgr-0c8e94728bc9ff5b	SSH	TCP	22	Pesso...	Acesso remoto
sgr-004d45e7f3aafb079	TCP personalizado	TCP	111	Pesso...	
sgr-036216d0cc90a1928	NFS	TCP	2049	Pesso...	

Fonte: Elaborado Pelo Grupo

Para verificar se as portas que foram adicionadas estão funcionando é só usar o comando “**sudo netstat -tulnp | grep -E “2049|111”**“

Figura 60 – Verificando se as portas foram adicionadas e se estão funcionando



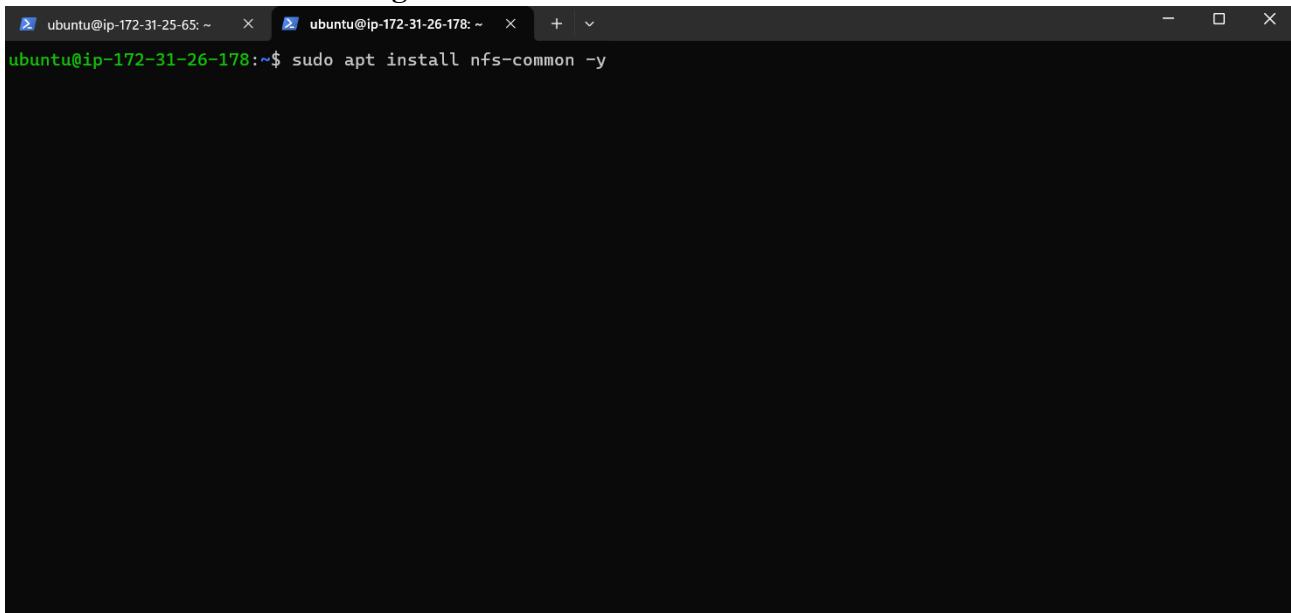
The screenshot shows two terminal windows side-by-side. The left window is titled 'ubuntu@ip-172-31-25-65: ~' and the right window is titled 'ubuntu@ip-172-31-26-178: ~'. Both windows display the command 'sudo netstat -tulnp | grep -E "2049|111"'. The output lists network ports:

```
ubuntu@ip-172-31-25-65:~$ sudo netstat -tulnp | grep -E "2049|111"
sudo: netstats: command not found
ubuntu@ip-172-31-25-65:~$ sudo netstat -tulnp | grep -E "2049|111"
tcp        0      0 0.0.0.0:2049          0.0.0.0:*
tcp        0      0 0.0.0.0:111         0.0.0.0:*
tcp6       0      0 :::2049            :::*                LISTEN      -
tcp6       0      0 :::111             :::*                LISTEN      1/init
udp        0      0 0.0.0.0:111         0.0.0.0:*
udp6       0      0 :::111             :::*                LISTEN      1/init
ubuntu@ip-172-31-25-65:~$
```

Fonte: Elaborado Pelo Grupo

Agora vamos configurar a máquina do outro lado, no caso a do cliente. Começamos baixando o “**nfs-common**” na máquina cliente:

Figura 61 – Instalando o nfs-common



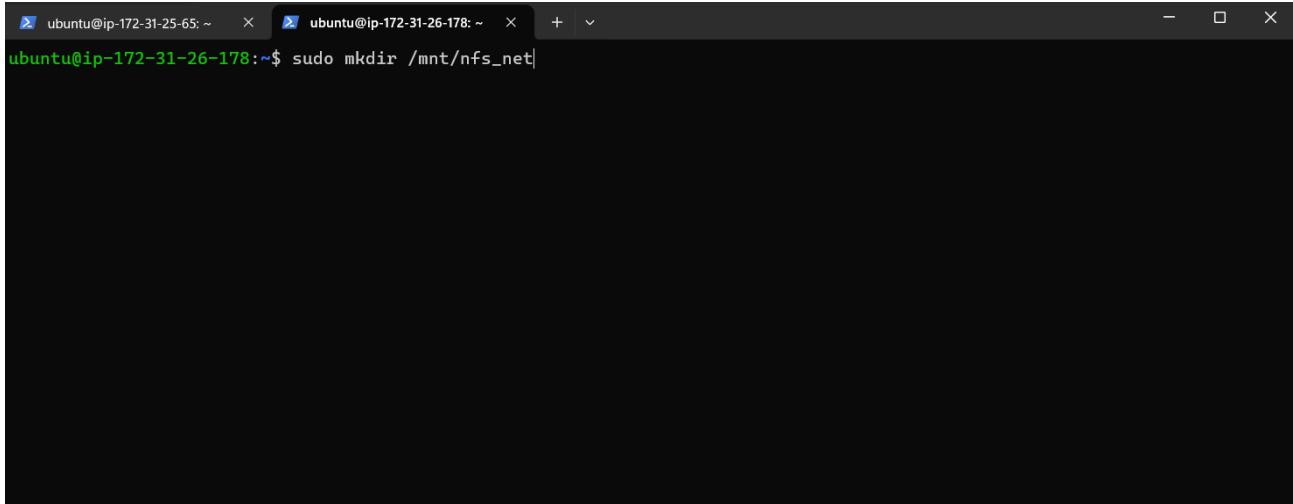
The screenshot shows a single terminal window titled 'ubuntu@ip-172-31-26-178: ~'. It displays the command 'sudo apt install nfs-common -y'.

```
ubuntu@ip-172-31-26-178:~$ sudo apt install nfs-common -y
```

Fonte: Elaborado Pelo Grupo

Depois de baixado criamos um diretório dentro do repositório “**/mnt**” com o comando “**sudo mkdir /mnt/nfs_net**”

Figura 62 – Criando repositório nfs_net



```
ubuntu@ip-172-31-25-65: ~ x 2 ubuntu@ip-172-31-26-178: ~ x + v
ubuntu@ip-172-31-26-178:~$ sudo mkdir /mnt/nfs_net|
```

Fonte: Elaborado Pelo Grupo

Em seguida damos permissão a esse diretório com o comando “**sudo chmod 777 mnt/nfs_net/**”

Figura 63– Dando permissão total pro diretório

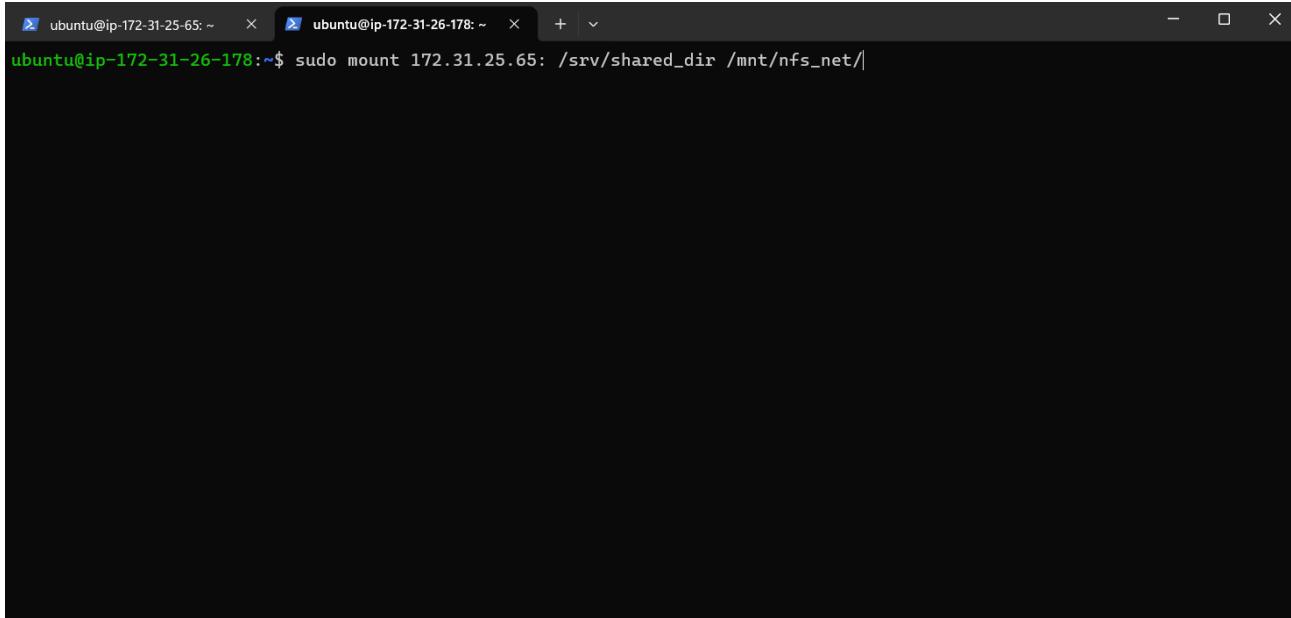


```
ubuntu@ip-172-31-25-65: ~ x 2 ubuntu@ip-172-31-26-178: ~ x + v
ubuntu@ip-172-31-26-178:~$ sudo chmod 777 mnt/nfs_net/|
```

Fonte: Elaborado Pelo Grupo

E agora vamos montar um ponto dentro desse diretório que foi criado, usando o IP da máquina servidor usando o comando “**sudo mount 172.31.25.65:/srv/shared_dir mnt/nfs_net/**”

Figura 64 – Criando um ponto no diretório

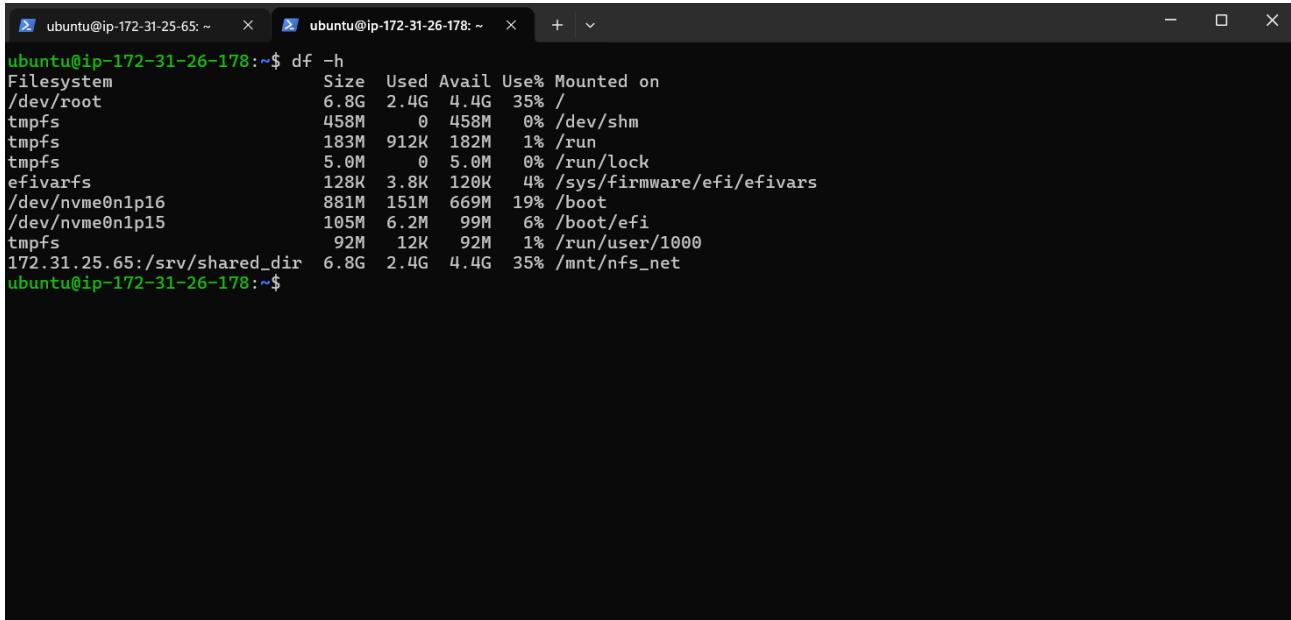


```
ubuntu@ip-172-31-26-178:~$ sudo mount 172.31.25.65:/srv/shared_dir /mnt/nfs_net/
```

Fonte: Elaborado Pelo Grupo

Para testar se não deu nenhum erro é só usar o comando “**df -h**”.

Figura 65 – Testando para ver se não deu erro

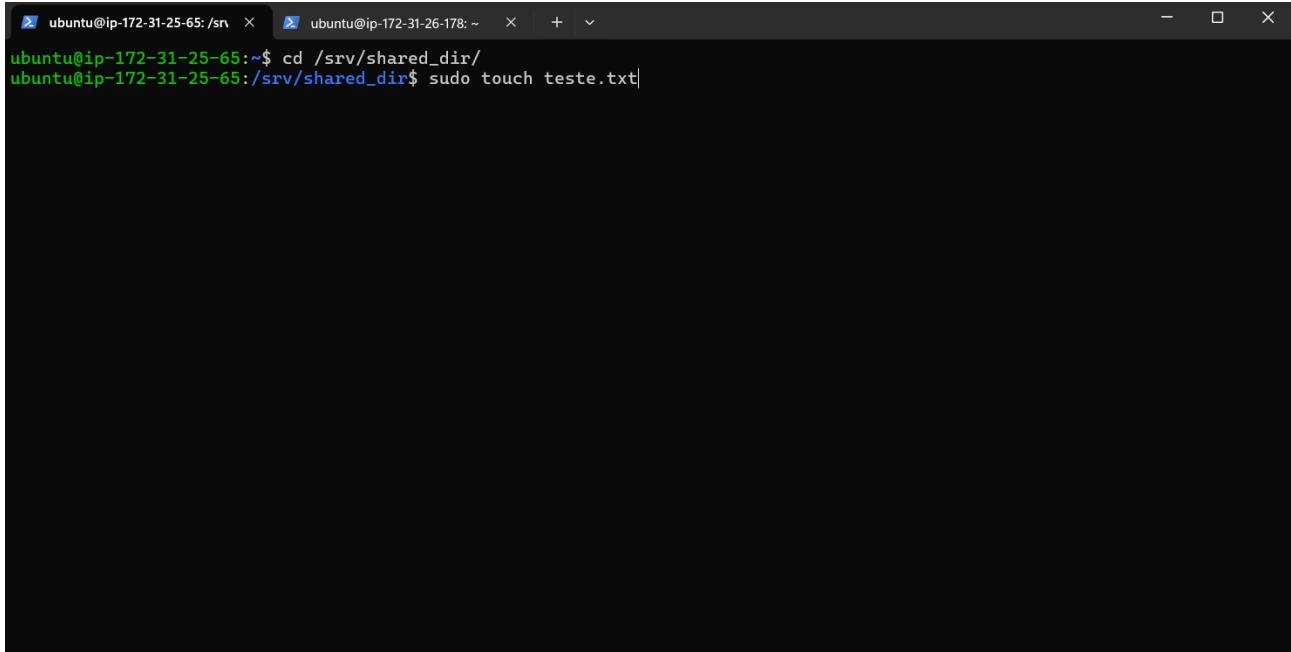


```
ubuntu@ip-172-31-26-178:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       6.8G  2.4G  4.4G  35% /
tmpfs          458M   0  458M   0% /dev/shm
tmpfs          183M  912K  182M   1% /run
tmpfs          5.0M   0  5.0M   0% /run/lock
efivarfs        128K  3.8K  120K   4% /sys/firmware/efi/efivars
/dev/nvme0n1p6  881M 151M  669M  19% /boot
/dev/nvme0n1p15 105M  6.2M  99M   6% /boot/efi
tmpfs           92M   12K  92M   1% /run/user/1000
172.31.25.65:/srv/shared_dir  6.8G  2.4G  4.4G  35% /mnt/nfs_net
ubuntu@ip-172-31-26-178:~$
```

Fonte: Elaborado Pelo Grupo

Agora vamos testar na prática entrando nesse diretório com o comando “**cd /mnt/nfs_net**” na máquina cliente e na máquina servidor usamos o “**cd /srv/shared_dir/**” e então usamos o “**sudo touch teste.txt**”.

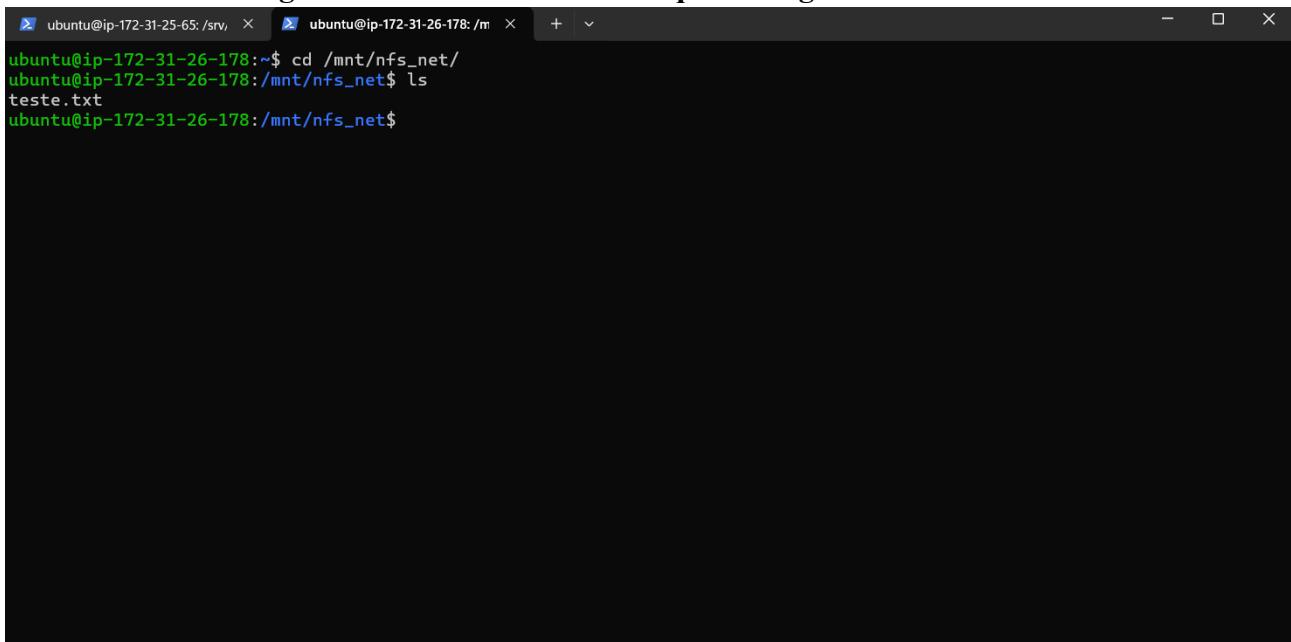
Figura 66 – Criando um arquivo teste no servidor



```
ubuntu@ip-172-31-25-65:~/srv$ cd /srv/shared_dir/
ubuntu@ip-172-31-25-65:/srv/shared_dir$ sudo touch teste.txt
```

Fonte: Elaborado Pelo Grupo

Figura 67 – Conferindo se o arquivo chegou no cliente



```
ubuntu@ip-172-31-26-178:~/mnt/nfs_net$ ls
teste.txt
ubuntu@ip-172-31-26-178:/mnt/nfs_net$
```

Fonte: Elaborado Pelo Grupo

Pronto, o setup Network File System está completo.fewfwf

1.1.5 Domain Name System(DNS)

O DNS, sigla para Domain Name System (Sistema de Nomes de Domínio), é um dos pilares fundamentais da internet. Sua principal função é atuar como uma "lista telefônica da web", traduzindo os nomes de domínio que são fáceis de ler e memorizar para humanos (como www.google.com) nos endereços de IP numéricos (como 111.111.111.11) que as máquinas usam

para se identificar e se comunicar na rede. Quando um usuário digita o endereço de um site em seu navegador, uma consulta é enviada a um servidor DNS. Este servidor localiza o endereço IP correspondente àquele domínio e o devolve ao navegador, permitindo que a conexão com o servidor do site seja estabelecida. Sem o DNS, teríamos que memorizar sequências complexas de números para cada site que quiséssemos visitar, o que tornaria a navegação na internet inviável. Portanto, ele é um sistema essencial que organiza e facilita o acesso a recursos online de forma transparente e eficiente.

Foi criado na AWS, duas instâncias:

- **Instância do servidor DNS.**
- **Instância do cliente.**

Figura 68 – Console AWS DNS

Instâncias (2) Informações		Última atualização 6 minutos atrás		Conectar	Estado da instância	Ações	Executar instâncias
	Name	ID da instância	Todos os ...				
<input type="checkbox"/>	Servidor	i-0e8ab1496ad567fb2	Executando	t2.micro	2/2 verificações a Exibir alarmes +	us-east-1d	ec2-54-147-51-5.comp... 54.147.51.5 -
<input type="checkbox"/>	Cliente DNS	i-059c214b1434db2b0	Executando	t2.micro	2/2 verificações a Exibir alarmes +	us-east-1d	ec2-34-197-71-73.com... 34.197.71.73 34.197.71.73

Fonte: Elaborado Pelo Grupo

Este é o servidor DNS principal: **Servidor Bind9 (172.31.30.87)**. Ele foi configurado para resolver nomes dentro do domínio privado “soloforteagro.teste” e também fazer a resolução reversa (IP → nome).

Figura 69 - Verificação se o serviço Bind9 está rodando:

```
ubuntu@ip-172-31-30-87:~$ sudo systemctl status named
● named.service - BIND Domain Name Server
  Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; preset: enabled)
  Active: active (running) since Fri 2025-10-17 21:27:30 UTC; 19min ago
    Docs: man:named(8)
   Main PID: 530 (named)
      Status: "running"
        Tasks: 5 (limit: 1121)
       Memory: 31.0M (peak: 31.2M)
          CPU: 67ms
         CGroup: /system.slice/named.service
                   └─530 /usr/sbin/named -f -u bind

Oct 17 21:27:30 ip-172-31-30-87 named[530]: network unreachable resolving './NS/IN': 2001:7fe::53#53
Oct 17 21:27:30 ip-172-31-30-87 named[530]: network unreachable resolving './NS/IN': 2001:503:ba3e::2>
Oct 17 21:27:30 ip-172-31-30-87 named[530]: network unreachable resolving './NS/IN': 2001:503:c27::2>
Oct 17 21:27:30 ip-172-31-30-87 named[530]: network unreachable resolving './NS/IN': 2001:500:a8::e#53
Oct 17 21:27:30 ip-172-31-30-87 named[530]: network unreachable resolving './NS/IN': 2001:500:2f::f#53
Oct 17 21:27:30 ip-172-31-30-87 named[530]: network unreachable resolving './NS/IN': 2001:dc3::35#53
Oct 17 21:27:30 ip-172-31-30-87 named[530]: network unreachable resolving './NS/IN': 2001:500:2::c#53
Oct 17 21:27:30 ip-172-31-30-87 named[530]: network unreachable resolving './NS/IN': 2001:7fd::1#53
Oct 17 21:27:30 ip-172-31-30-87 named[530]: managed-keys-zone: Key 20326 for zone . is now trusted (a>
Oct 17 21:27:30 ip-172-31-30-87 named[530]: managed-keys-zone: Key 38696 for zone . is now trusted (a>
lines 1-22/22 (END)
```

Fonte: Elaborado Pelo Grupo

Este comando verifica o status do serviço named (processo principal do Bind9).

O objetivo é confirmar que o serviço foi iniciado corretamente e está ativado, sem erros. Conclusão:

- o Bind9 está em execução e pronto para responder às consultas DNS. O comando “**“named-checkzone”** valida os arquivos de zona direta (db.soloforteagro.teste) e reversa.

Figura 70 - Verificação se as zonas foram carregadas corretamente:

```
ubuntu@ip-172-31-30-87:~$ sudo named-checkzone soloforteagro.teste /etc/bind/db.soloforteagro.teste
sudo named-checkzone 30.31.172.in-addr.arpa /etc/bind/db.172.31.30
zone soloforteagro.teste/IN: loaded serial 2
OK
zone 30.31.172.in-addr.arpa/IN: loaded serial 1
OK
ubuntu@ip-172-31-30-87:~$ |
```

Fonte: Elaborado Pelo Grupo

Conclusão: as zonas foram carregadas sem erros e estão ativas no servidor DNS.

“**“@localhost”** indica que o teste está sendo feito diretamente contra o servidor Bind9 local. O primeiro comando testa o nome principal (soloforteagro.teste).

Figura 71 - Testando resolução local no próprio servidor:

```
ubuntu@ip-172-31-30-87:~$ dig @localhost soloforteagro.teste
; <>> DIG 9.18.39-0ubuntu0.24.04.1-Ubuntu <>> @localhost soloforteagro.teste
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<- opcode: QUERY, status: NOERROR, id: 42463
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 767e84777cbd328f0100000068f2bf21abf2dc4221e8dc12 (good)
;; QUESTION SECTION:
;soloforteagro.teste.      IN      A

;; ANSWER SECTION:
soloforteagro.teste.  604800  IN      A      172.31.30.87

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(localhost) (UDP)
;; WHEN: Fri Oct 17 22:11:45 UTC 2025
;; MSG SIZE  rcvd: 92
ubuntu@ip-172-31-30-87:~$ |
```

Fonte: Elaborado Pelo Grupo

Figura 72 - O segundo comando testa o subdomínio www:

```
ubuntu@ip-172-31-30-87:~$ dig @localhost www.soloforteagro.teste
; <>> DIG 9.18.39-0ubuntu0.24.04.1-Ubuntu <>> @localhost www.soloforteagro.teste
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<- opcode: QUERY, status: NOERROR, id: 34027
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

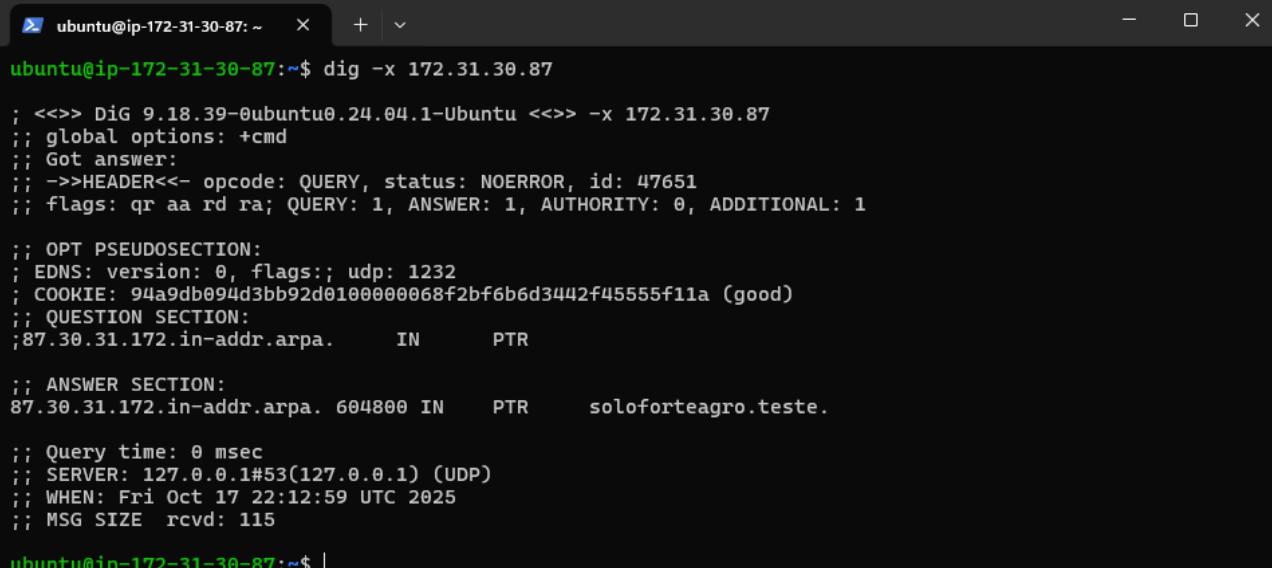
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 2c82bc5dae2e93b70100000068f2bf49d96eba6e3e6b4f91 (good)
;; QUESTION SECTION:
;www.soloforteagro.teste.      IN      A

;; ANSWER SECTION:
www.soloforteagro.teste.  604800  IN      A      172.31.30.87

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(localhost) (UDP)
;; WHEN: Fri Oct 17 22:12:25 UTC 2025
;; MSG SIZE  rcvd: 96
ubuntu@ip-172-31-30-87:~$ |
```

Fonte: Elaborado Pelo Grupo

Figura 73 - O terceiro faz a resolução reversa (PTR), convertendo o IP de volta em nome de domínio:



```
ubuntu@ip-172-31-30-87:~$ dig -x 172.31.30.87

; <>> Dig 9.18.39-0ubuntu0.24.04.1-Ubuntu <>> -x 172.31.30.87
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47651
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags: udp: 1232
;; COOKIE: 94a9db094d3bb92d0100000068f2bf6b6d3442f45555f11a (good)
;; QUESTION SECTION:
;87.30.31.172.in-addr.arpa. IN PTR soloforteagro.teste.

;; ANSWER SECTION:
87.30.31.172.in-addr.arpa. 604800 IN PTR soloforteagro.teste.

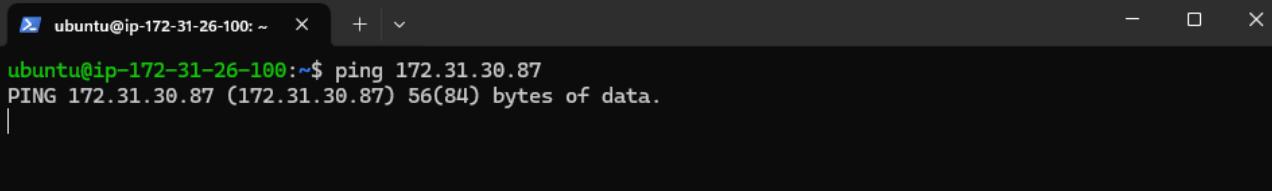
;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1) (UDP)
;; WHEN: Fri Oct 17 22:12:59 UTC 2025
;; MSG SIZE rcvd: 115

ubuntu@ip-172-31-30-87:~$ |
```

Fonte: Elaborado Pelo Grupo

Conclusão: o servidor DNS responde corretamente a consultas diretas e reversas internamente. **Agora teste com o Cliente DNS (172.31.26.100).** O objetivo dessa etapa é demonstrar que o servidor DNS configurado na instância anterior funciona corretamente para outras máquinas da rede. Esse comando verifica se a instância cliente consegue alcançar o servidor DNS via rede interna (VPC).

Figura 74 - Testar conectividade entre cliente e servidor:

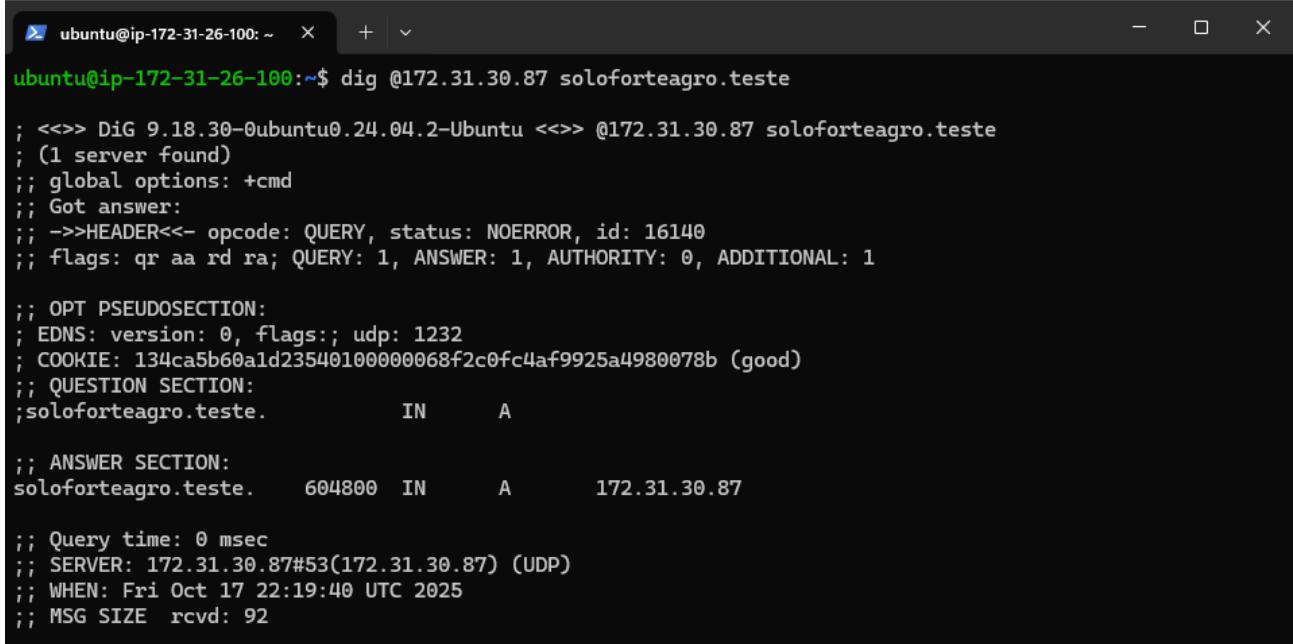


```
ubuntu@ip-172-31-26-100:~$ ping 172.31.30.87
PING 172.31.30.87 (172.31.30.87) 56(84) bytes of data.
```

Fonte: Elaborado Pelo Grupo

Conclusão esperada: as instâncias estão se comunicando dentro da mesma rede privada da AWS. Testar resolução do domínio e subdomínio: O que faz: envia uma consulta direta ao servidor DNS (172.31.30.87); Testa se o domínio e o subdomínio estão sendo resolvidos corretamente; Conclusão esperada: O cliente consegue resolver nomes definidos no Bind9 do servidor, confirmando que o DNS funciona externamente.

Figura 75 - Testar resolução do domínio e subdomínio



```
ubuntu@ip-172-31-26-100:~$ dig @172.31.30.87 soloforteagro.teste

; <>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <>> @172.31.30.87 soloforteagro.teste
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16140
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 134ca5b60a1d23540100000068f2c0fc4af9925a4980078b (good)
;; QUESTION SECTION:
;soloforteagro.teste.      IN      A

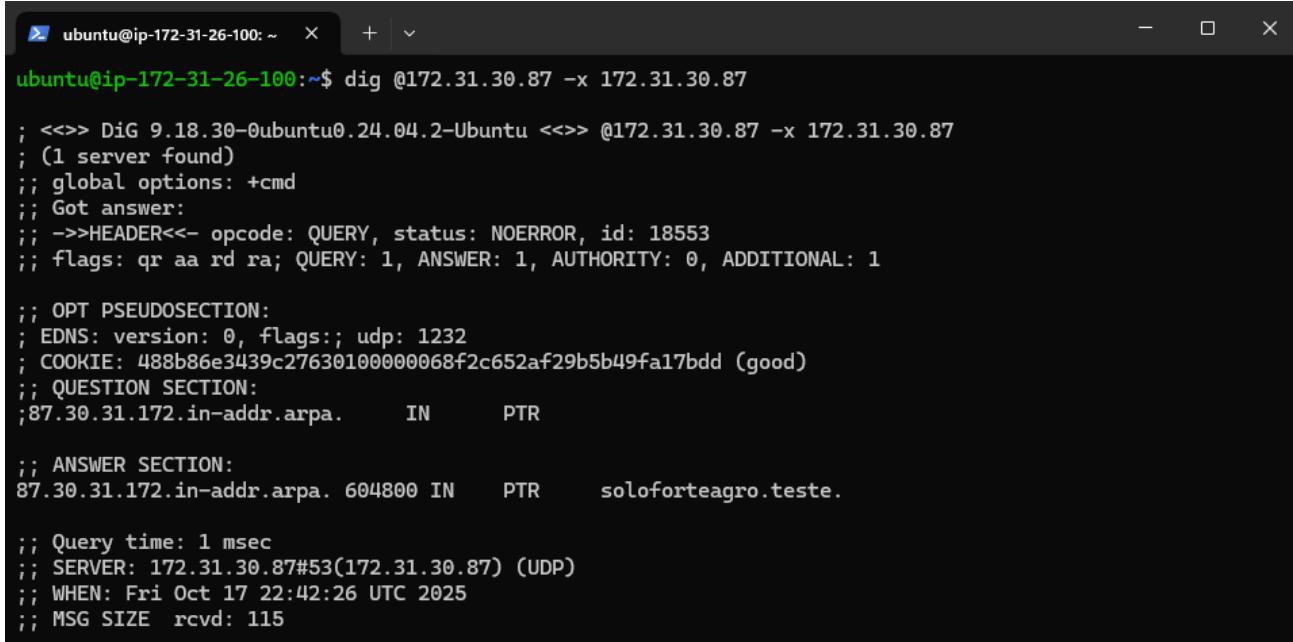
;; ANSWER SECTION:
soloforteagro.teste.  604800  IN      A      172.31.30.87

;; Query time: 0 msec
;; SERVER: 172.31.30.87#53(172.31.30.87) (UDP)
;; WHEN: Fri Oct 17 22:19:40 UTC 2025
;; MSG SIZE  rcvd: 92
```

Fonte: Elaborado Pelo Grupo

Testar resolução reversa: Realiza uma consulta reversa (IP → nome), usando a zona in-addr.arpa configurada no servidor. Conclusão esperada: O Bind9 responde corretamente com o nome associado ao IP, provando que a **resolução reversa está funcional**.

Figura 76 - Testar resolução reversa:



```
ubuntu@ip-172-31-26-100:~$ dig @172.31.30.87 -x 172.31.30.87

; <>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <>> @172.31.30.87 -x 172.31.30.87
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18553
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 488b86e3439c27630100000068f2c652af29b5b49fa17bdd (good)
;; QUESTION SECTION:
;87.30.31.172.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
87.30.31.172.in-addr.arpa.  604800  IN      PTR      soloforteagro.teste.

;; Query time: 1 msec
;; SERVER: 172.31.30.87#53(172.31.30.87) (UDP)
;; WHEN: Fri Oct 17 22:42:26 UTC 2025
;; MSG SIZE  rcvd: 115
```

Fonte: Elaborado Pelo Grupo

1.2 Ambiente On-Premise(Virtualbox)

Nesta seção, são detalhados os serviços configurados em um ambiente local (on-premise), utilizando máquinas virtuais gerenciadas pelo Oracle VM VirtualBox. Este ambiente simula uma infraestrutura interna da empresa, permitindo a configuração e o teste de serviços essenciais de rede de forma isolada e controlada.

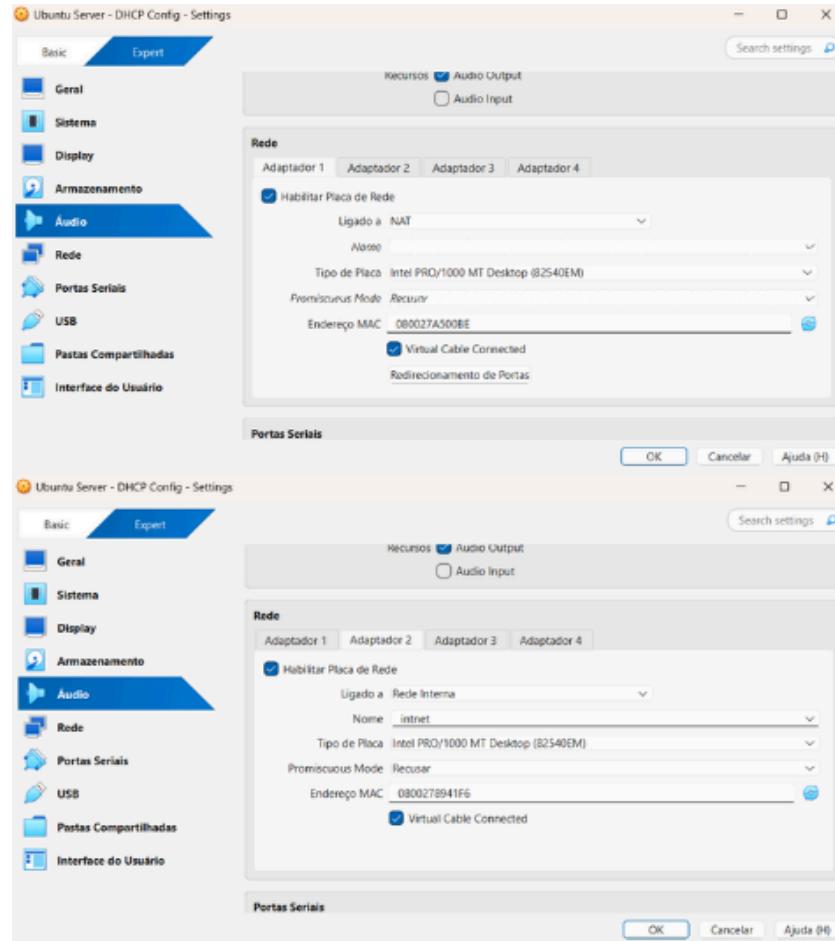
1.2.1 Serviço de DHCP

O protocolo DHCP (Dynamic Host Configuration Protocol, ou Protocolo de Configuração Dinâmica de Host) é um serviço fundamental em uma rede de computadores. Sua principal função é automatizar a atribuição de endereços IP e outras configurações de rede necessárias para que os dispositivos (clientes) possam se comunicar. Em vez de configurar manualmente cada máquina com um IP, máscara de sub-rede, gateway e servidores DNS, o servidor DHCP centraliza e gerencia essa distribuição, entregando as informações de forma automática a cada cliente que se conecta à rede.

Para este projeto, o serviço DHCP foi configurado em um servidor Ubuntu Server 22.04 e testado por um cliente Windows Server, ambos virtualizados no VirtualBox. O passo a passo da implementação é descrito a seguir.

O primeiro passo consiste em garantir que a máquina virtual do servidor possua duas interfaces de rede com funções distintas. A primeira (Adaptador 1) foi configurada em modo NAT, permitindo que a máquina virtual acesse a internet para baixar pacotes e atualizações. A segunda (Adaptador 2) foi configurada em modo Rede Interna (intnet), criando uma rede local isolada onde o servidor DHCP atuará, conectando-se futuramente à máquina cliente.

Figura 77 - Configuração de rede da máquina servidora no VirtualBox



Fonte: Elaborado Pelo Grupo

Para que o servidor DHCP possa operar, sua interface de rede interna precisa ter um endereço IP estático. A configuração foi realizada editando o arquivo de plano de rede (`/etc/netplan/00-installer-config.yaml`). Na interface de rede interna (`enp0s8`), a opção “**dhcp4**” foi definida como “**false**” e um endereço estático (192.168.99.1/24) foi atribuído.

Figura 78 - Arquivo de configuração Netplan com IP estático

```
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: true
    enp0s8:
      dhcp4: false
      addresses: [192.168.99.1/24]
      gateway4: 10.0.2.15
      nameservers:
        addresses: [8.8.8.8, 8.8.4.4]
guigs@dhcp-config:~$ _
```

Fonte: Elaborado Pelo Grupo

Após salvar a configuração, o comando “**sudo netplan apply**” foi executado para aplicar as alterações.

Com a rede configurada, o pacote do servidor DHCP, “**isc-dhcp-server**”, foi instalado no Ubuntu Server utilizando o gerenciador de pacotes “**apt**”.

Figura 79 - Comando de instalação do pacote isc-dhcp-server

```
guigs@dhcp-config:~$ sudo apt install isc-dhcp-server -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
isc-dhcp-server is already the newest version (4.4.3-P1-4ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
guigs@dhcp-config:~$ _
```

Fonte: Elaborado Pelo Grupo

O arquivo principal de configuração do DHCP, /etc/dhcp/dhcpd.conf, foi editado para definir o escopo (range) de endereços que seriam distribuídos aos clientes. Foi definida uma sub-rede (subnet) 192.168.1.0 com máscara 255.255.255.0. Dentro dela, foram especificados: range: O intervalo de IPs a serem distribuídos (de 192.168.1.51 a 192.168.1.100). Option routers: O gateway padrão para os clientes (o próprio servidor, 192.168.1.1). Option domain-name-servers: Os servidores DNS a serem utilizados pelos clientes (8.8.8.8 e 1.1.1.1).

Figura 80 - Configuração da sub-rede e escopo no arquivo dhcpcd.conf

```
#shared-network 224-29 {
#  subnet 10.17.224.0 netmask 255.255.255.0 {
#    option routers rtr-224.example.org;
#  }
#  subnet 10.0.29.0 netmask 255.255.255.0 {
#    option routers rtr-29.example.org;
#  }
#  pool {
#    allow members of "foo";
#    range 10.17.224.10 10.17.224.250;
#  }
#  pool {
#    deny members of "foo";
#    range 10.0.29.10 10.0.29.230;
#  }
#}
subnet 192.168.99.0 netmask 255.255.255.0 {
  range 192.168.99.51 192.168.99.100;
  option routers 192.168.99.1;
  option domain-name-servers 8.8.8.8, 1.1.1.1;
  option domain-name "exemplo.org";
}
guigs@dhcp-config:~$
```

Fonte: Elaborado Pelo Grupo

Para garantir que o servidor DHCP operasse apenas na rede correta, o arquivo /etc/default/isc-dhcp-server foi editado, especificando no parâmetro “**INTERFACESv4**” o nome da interface de rede interna (enp0s8).

Figura 81 - Definição da interface de atuação do serviço DHCP

```
guigs@dhcp-config:~$ sudo cat /etc/default/isc-dhcp-server
# Defaults for isc-dhcp-server (sourced by /etc/init.d/isc-dhcp-server)

# Path to dhcpcd's config file (default: /etc/dhcp/dhcpcd.conf).
#DHCPDV4_CONF=/etc/dhcp/dhcpcd.conf
#DHCPDV6_CONF=/etc/dhcp/dhcpcd6.conf

# Path to dhcpcd's PID file (default: /var/run/dhcpcd.pid).
#DHCPDV4_PID=/var/run/dhcpcd.pid
#DHCPDV6_PID=/var/run/dhcpcd6.pid

# Additional options to start dhcpcd with.
#       Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

# On what interfaces should the DHCP server (dhcpcd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="enp0s8"
INTERFACESv6=""
guigs@dhcp-config:~$
```

Fonte: Elaborado Pelo Grupo

Após todas as configurações, o serviço foi reiniciado com o comando “**sudo service isc-dhcp-server restart**”. Em seguida, o status do serviço foi verificado com “**sudo service isc-dhcp-server status**” para confirmar que ele estava ativo e rodando sem erros.

Figura 82 - Verificação do status do serviço DHCP como ativo (running)

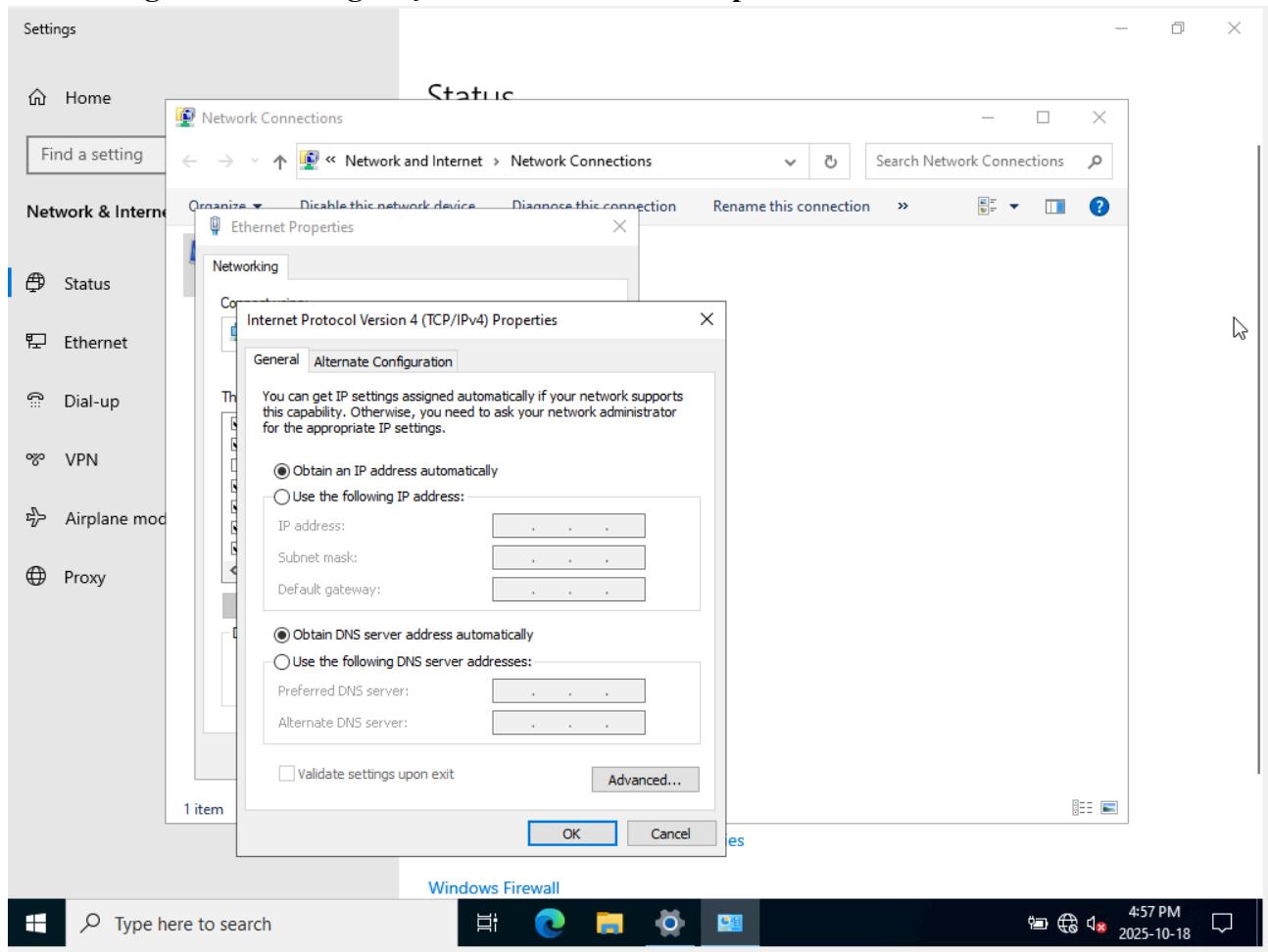
```
guigs@dhcp-config:~$ sudo service isc-dhcp-server status
● isc-dhcp-server.service - ISC DHCP IPv4 server
  Loaded: loaded (/usr/lib/systemd/system/isc-dhcp-server.service; enabled; preset: enabled)
  Active: active (running) since Sat 2025-10-18 19:09:50 UTC; 43min ago
    Docs: man:dhcpcd(8)
   Main PID: 868 (dhcpcd)
     Tasks: 1 (limit: 2268)
    Memory: 3.7M (peak: 4.0M)
      CPU: 16ms
     CGroup: /system.slice/isc-dhcp-server.service
             └─868 dhcpcd -user dhcpcd -group dhcpcd -f -4 -pf /run/dhcp-server/dhcpcd.pid -cf /etc/dhcp/dhcpcd.conf enp0s8

Oct 18 19:09:50 dhcp-config sh[868]: PID file: /run/dhcp-server/dhcpcd.pid
Oct 18 19:09:50 dhcpcd[868]: Wrote 0 leases to leases file.
Oct 18 19:09:50 dhcpcd[868]: Wrote 0 leases to leases file.
Oct 18 19:09:50 dhcpcd[868]: Listening on LPF/enp0s8/08:00:27:89:41:f6/192.168.99.0/24
Oct 18 19:09:50 dhcpcd[868]: Listening on LPF/enp0s8/08:00:27:89:41:f6/192.168.99.0/24
Oct 18 19:09:50 dhcpcd[868]: Sending on  LPF/enp0s8/08:00:27:89:41:f6/192.168.99.0/24
Oct 18 19:09:50 dhcpcd[868]: Sending on  LPF/enp0s8/08:00:27:89:41:f6/192.168.99.0/24
Oct 18 19:09:50 dhcpcd[868]: Sending on  Socket/fallback/fallback-net
Oct 18 19:09:50 dhcpcd[868]: Sending on  Socket/fallback/fallback-net
Oct 18 19:09:50 dhcpcd[868]: Server starting service.
guigs@dhcp-config:~$ _
```

Fonte: Elaborado Pelo Grupo

Para validar o funcionamento do servidor, uma máquina virtual com Windows Server foi configurada para atuar como cliente. Sua interface de rede foi definida para o mesmo modo Rede Interna (intnet) do servidor. Nas configurações do adaptador de rede IPv4, a máquina foi ajustada para obter um endereço IP e endereços de servidor DNS automaticamente.

Figura 83 - Configuração do cliente Windows para obter IP automaticamente



Fonte: Elaborado Pelo Grupo

No prompt de comando do cliente Windows, o comando “**ipconfig /all**” foi executado. A saída confirmou o sucesso da operação: o cliente recebeu o endereço IP 192.168.1.51, o primeiro disponível no escopo definido, além do gateway e dos servidores DNS corretos, validando que o servidor DHCP está funcionando como esperado.

Figura 84 - Saída do comando ipconfig /all no cliente, mostrando o IP recebido

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.20348.587]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Administrator>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

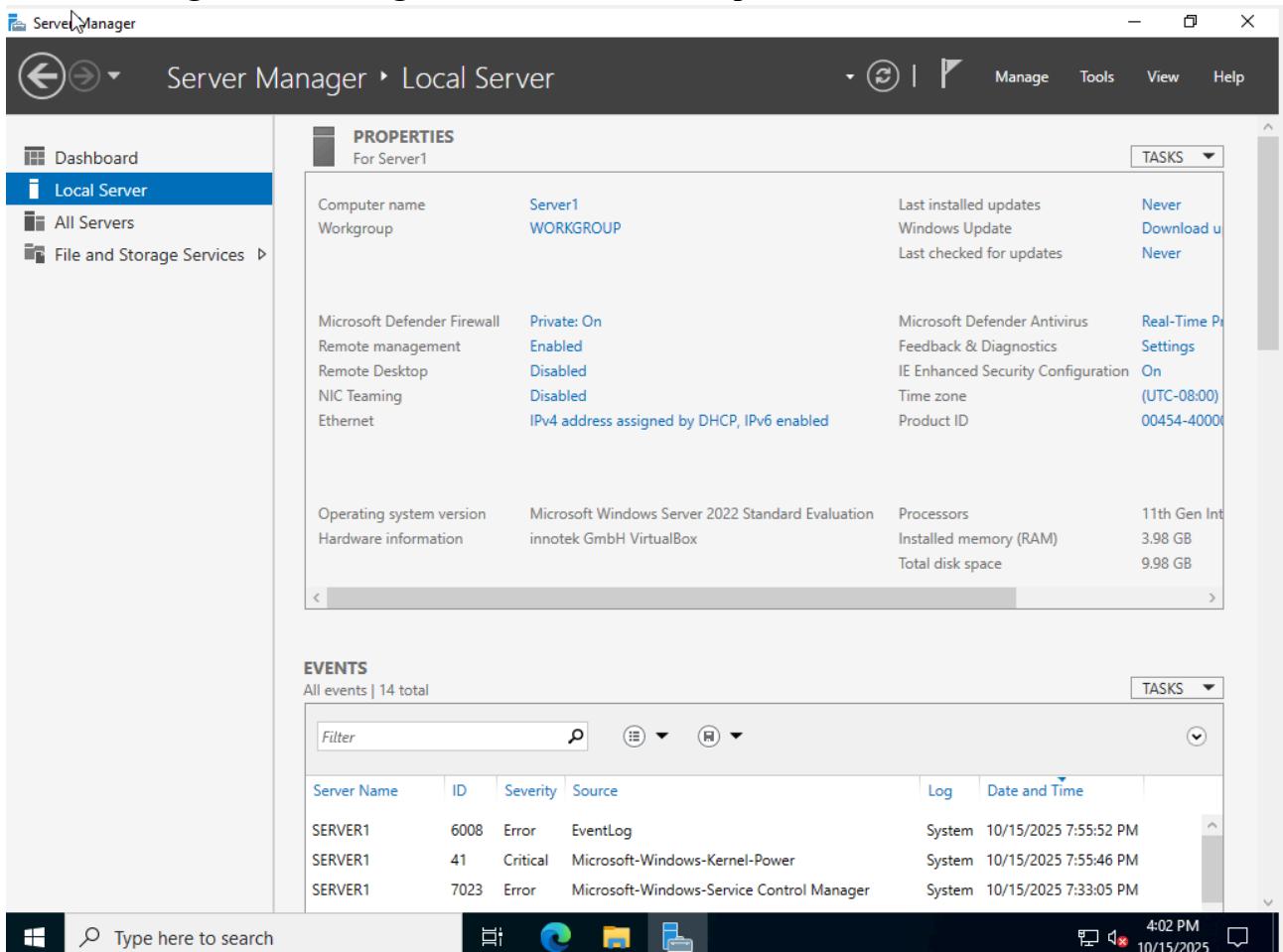
  Connection-specific DNS Suffix . : exemplo.org
  Link-local IPv6 Address . . . . . : fe80::3112:cad3:b627:e71c%7
  IPv4 Address . . . . . : 192.168.99.51
  Subnet Mask . . . . . : 255.255.255.0
  Default Gateway . . . . . : 192.168.99.1

C:\Users\Administrator>
```

Fonte: Elaborado Pelo Grupo

As configurações AD (Active Directory) e GPO (Group Policy Object) são ferramentas disponíveis para o gerenciamento da segurança em uma rede. Elas trabalham em conjunto para centralizar a configuração e segurança de usuários e máquinas. O AD organiza a rede em domínios e unidades organizacionais, enquanto os GPOs contêm configurações que podem ser aplicadas aos objetos contidos no AD através do Console de Gerenciamento de Políticas de Grupo (GPMC). Para este projeto, o AD e o GPO foram configurados em um servidor Windows virtualizado no VirtualBox. O passo a passo da implementação é descrito a seguir.

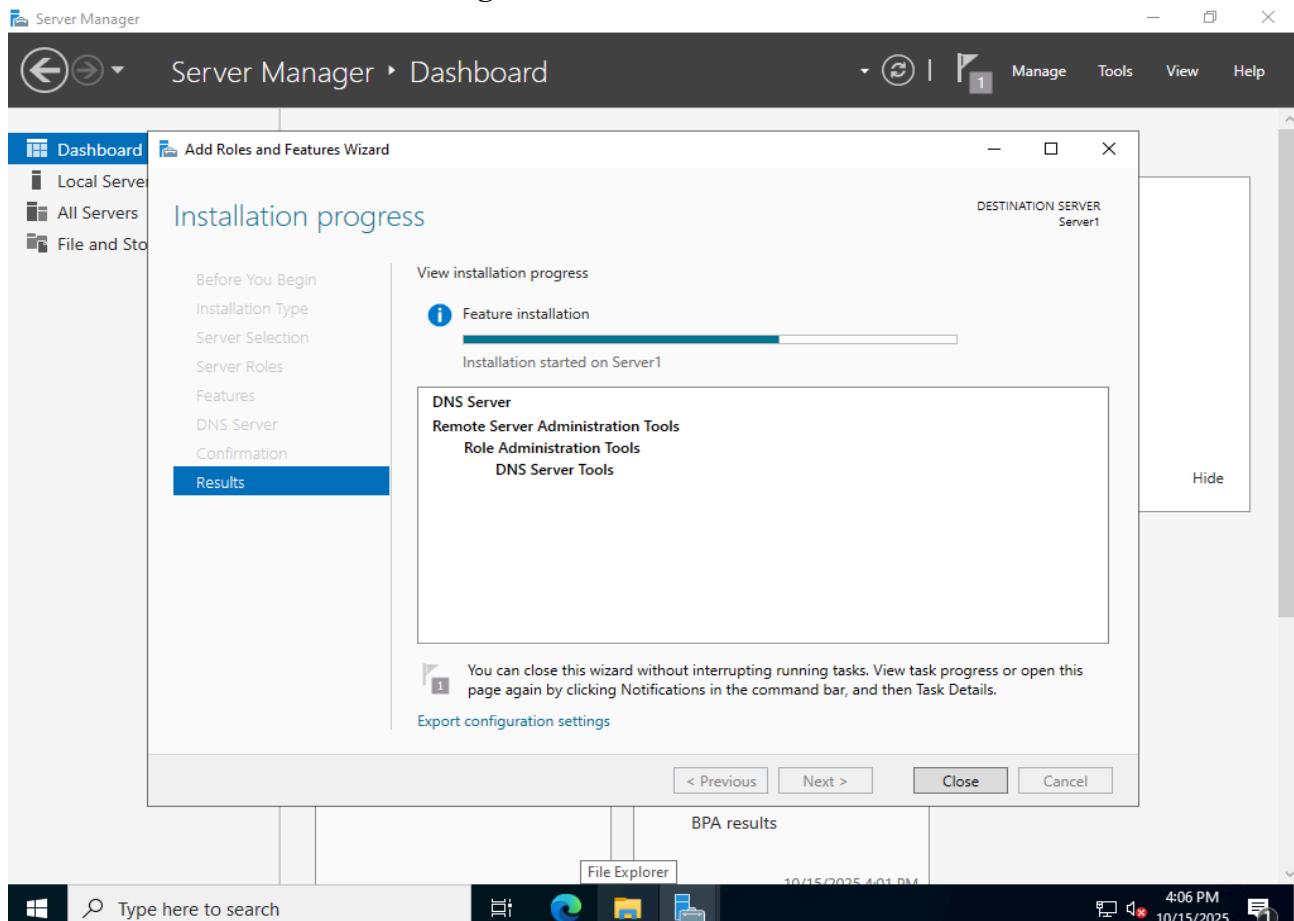
Para realizar esta promoção é necessário renomear o servidor que será promovido, abrindo o gerenciador de servidor e trocando o nome da máquina, junto a troca de nome do computador deve-se permitir as conexões remotas na aba “**Remoto**” nas propriedades do sistema onde também o nome foi modificado.

Figura 85 - Configuração do Server Local para controlador de domínio

Fonte: Elaborado Pelo Grupo

Após renomear o servidor inicia-se o processo de instalação do DNS e do AD. A aba Painel foi selecionada e por ela foi iniciado o processo de adição dos recursos AD e DNS clicando em adicionar funções e recursos, escolhendo a opção “instalação baseada em função ou recurso”, selecionando o servidor escolhido e marcando a opção “Servidor DNS”. Após isso inicia-se a instalação.

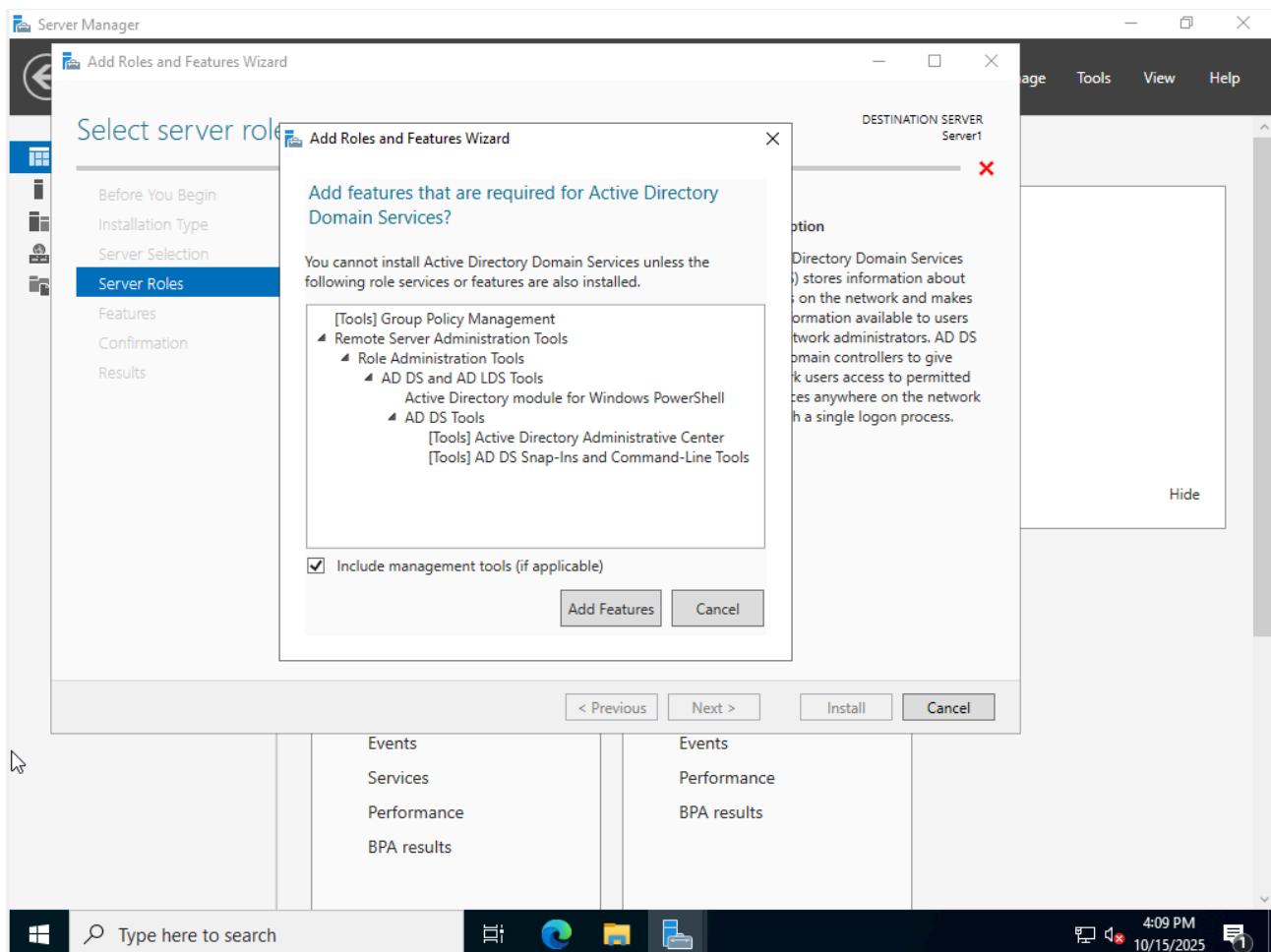
Figura 86 - instalando o DNS



Fonte: Elaborado Pelo Grupo

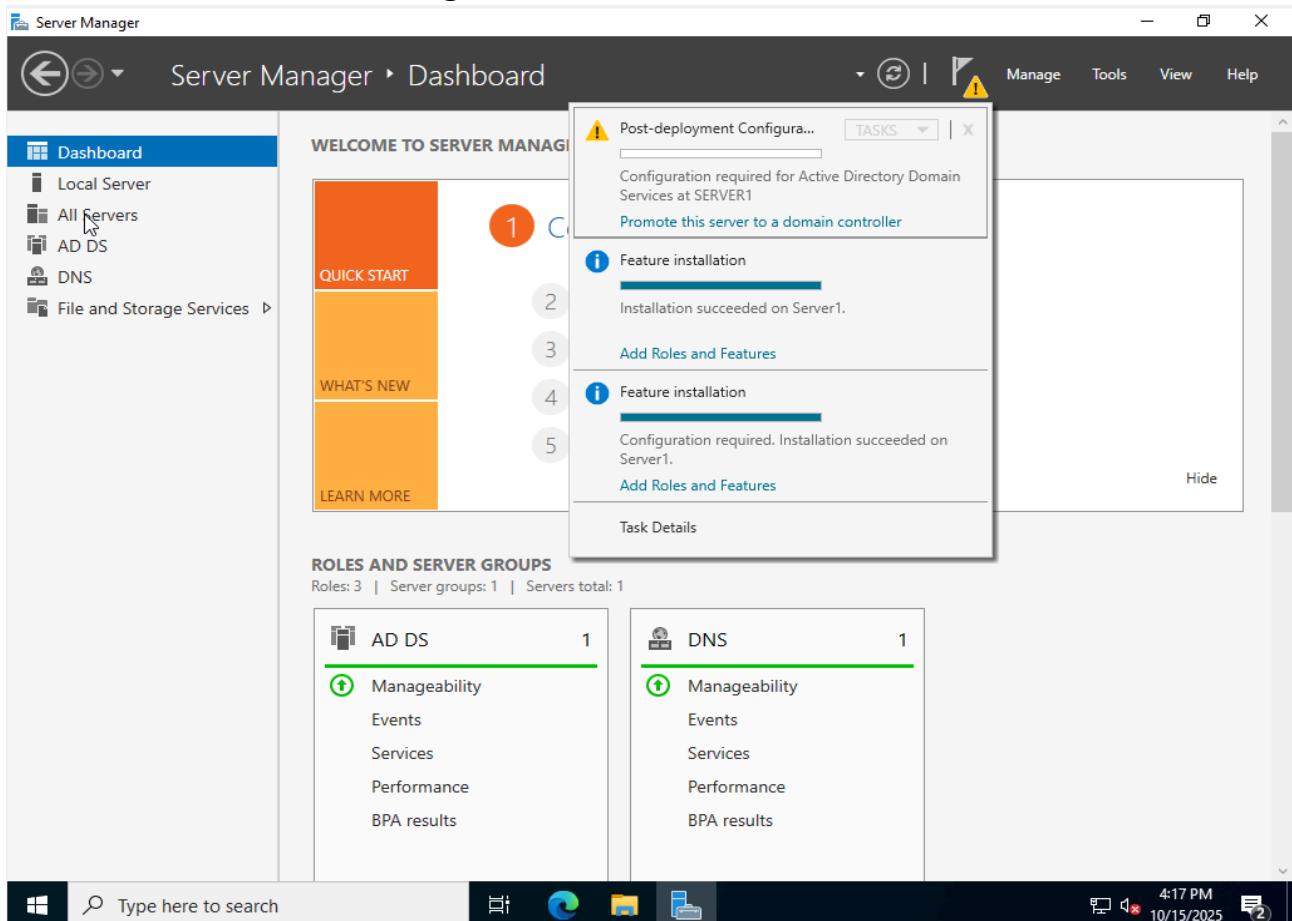
Agora é necessário repetir o processo porém nas funções do servidor deve-se selecionar a opção “Serviços de domínio Active Directory”

Figura 87 - instalando o AD



Fonte: Elaborado Pelo Grupo

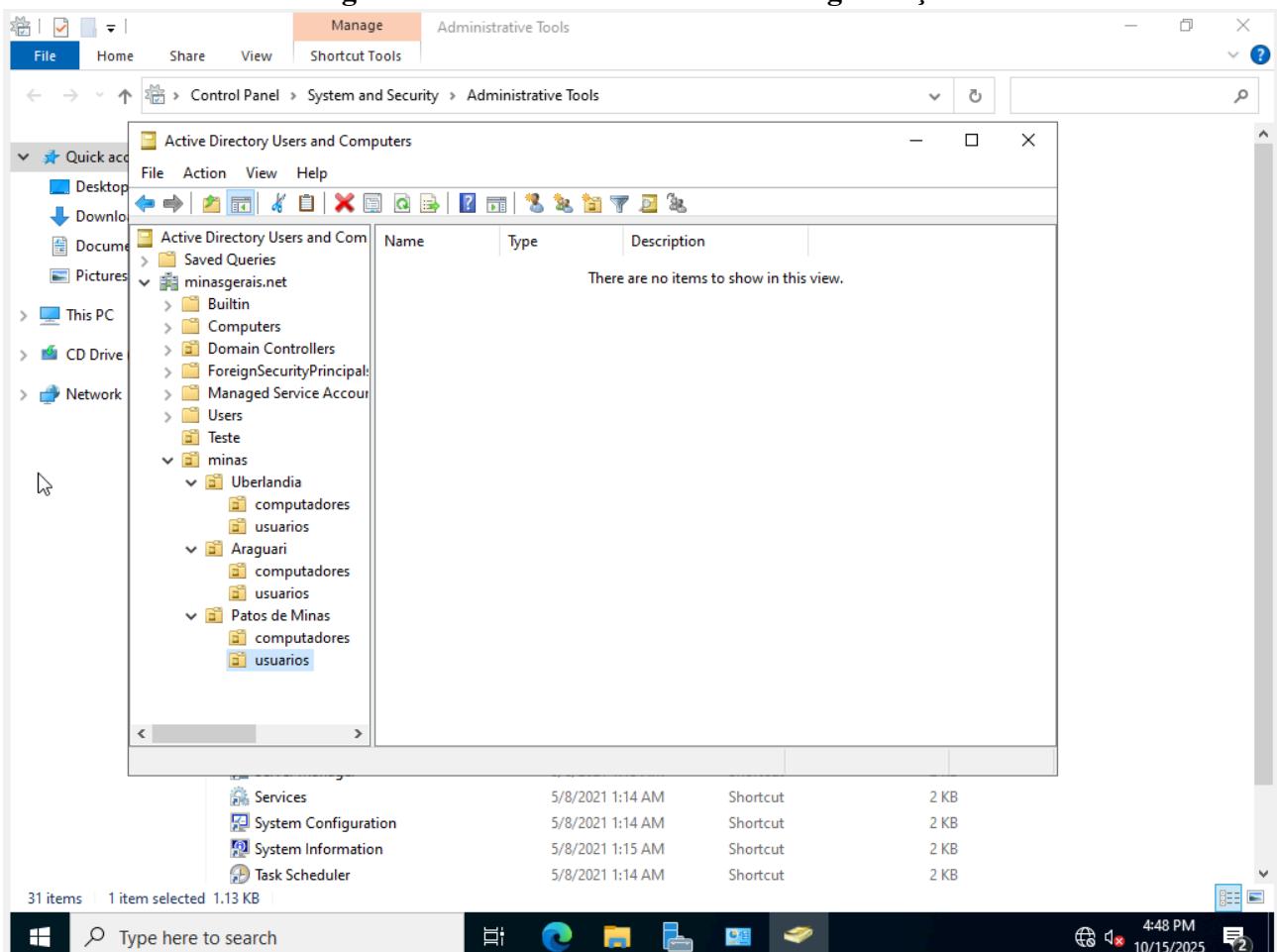
Com as configurações acima instaladas deve-se voltar ao painel e selecionar o símbolo de aviso ao lado da bandeira para iniciar a promoção do servidor a controlador de domínio.

Figura 88 - Promovendo servidor**Fonte: Elaborado Pelo Grupo**

A Microsoft projetou o AD para usar o DNS como sistema de nomeação, e DNS é hierárquico por natureza. O primeiro domínio do AD é chamado de raiz da árvore. Abaixo será mostrado como iniciar a configuração da raiz para o início da floresta. Na adição de uma nova floresta criaremos o domínio raiz nomeado de “minasgerais.net”. Prosseguimos com o passo a passo até verificar os requisitos necessários para o funcionamento, instalá-los e reiniciar o servidor.

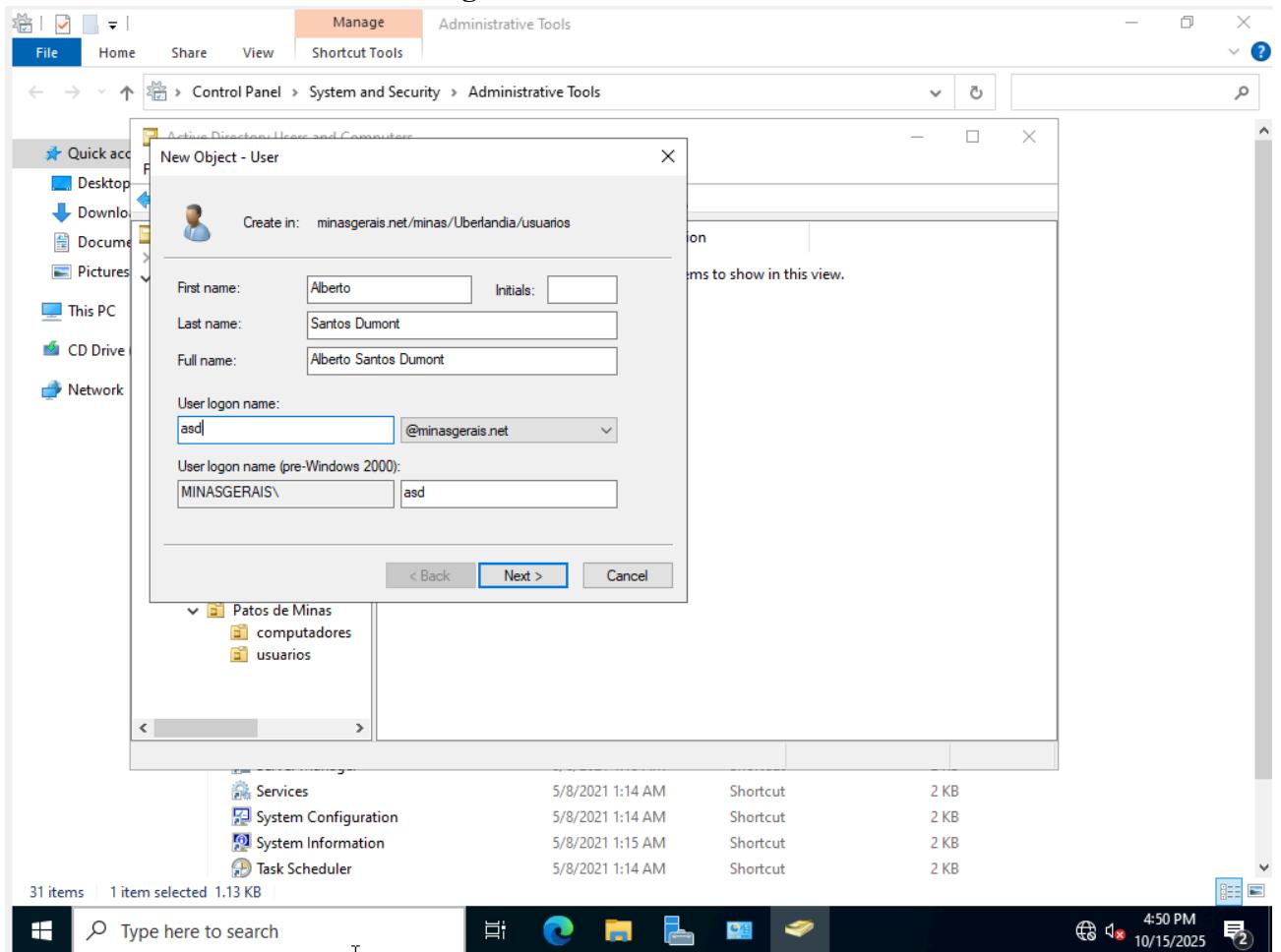
Após essa configuração inicial deve-se abrir as ferramentas administrativas, e selecionar “usuários e computadores do Active Directory” onde deve-se selecionar o domínio “minasgerais.net” e criar unidades organizacionais para cada filial onde serão colocados os computadores e os usuários.

Figura 89 - Criando as unidades de organização



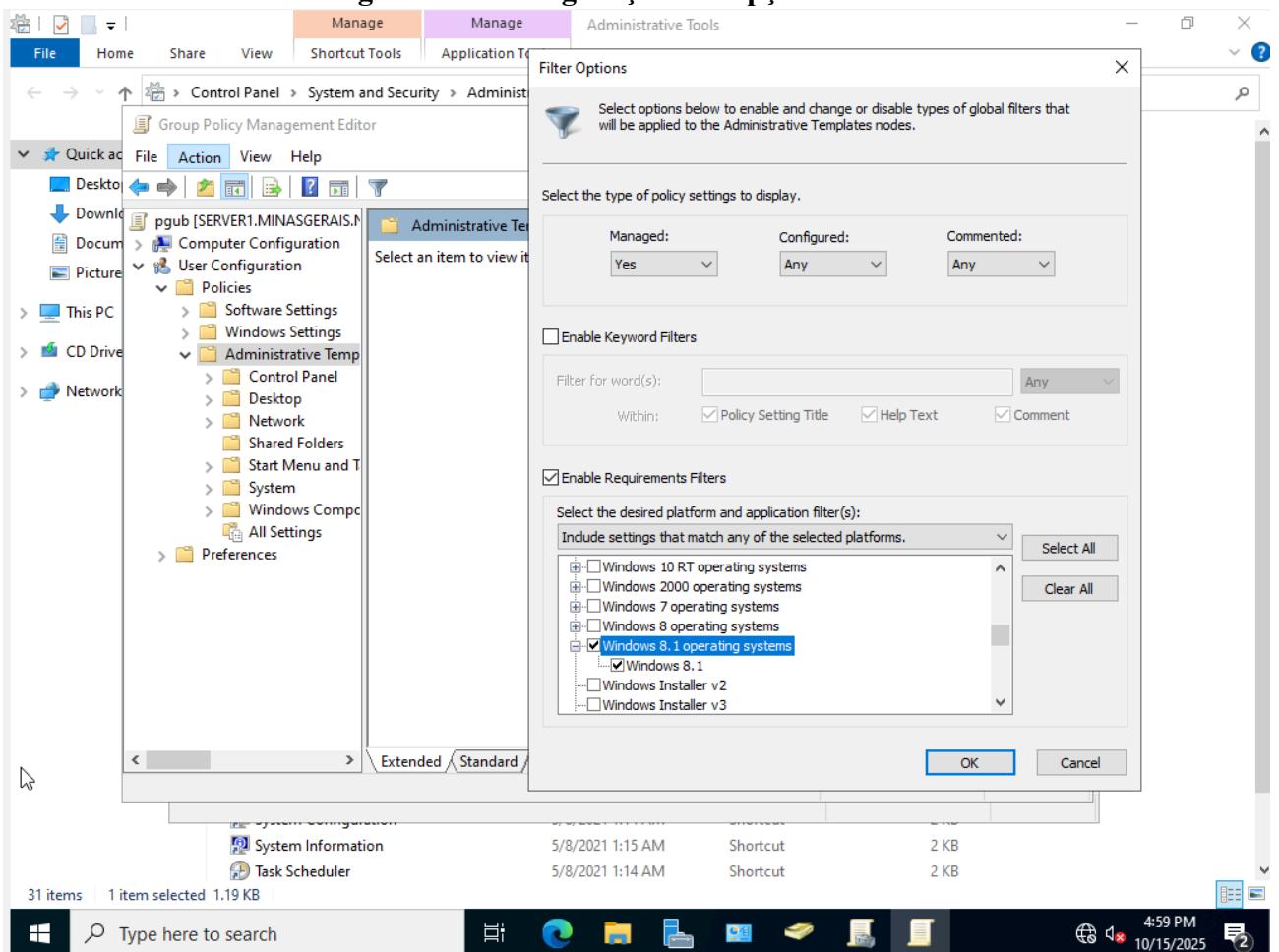
Fonte: Elaborado Pelo Grupo

Dentro de uma unidade organizacional foi criado o primeiro usuário com o nome fictício de Alberto Santos Dumont com a senha “puc@1958”.

Figura 90 - Criando o usuário**Fonte: Elaborado Pelo Grupo**

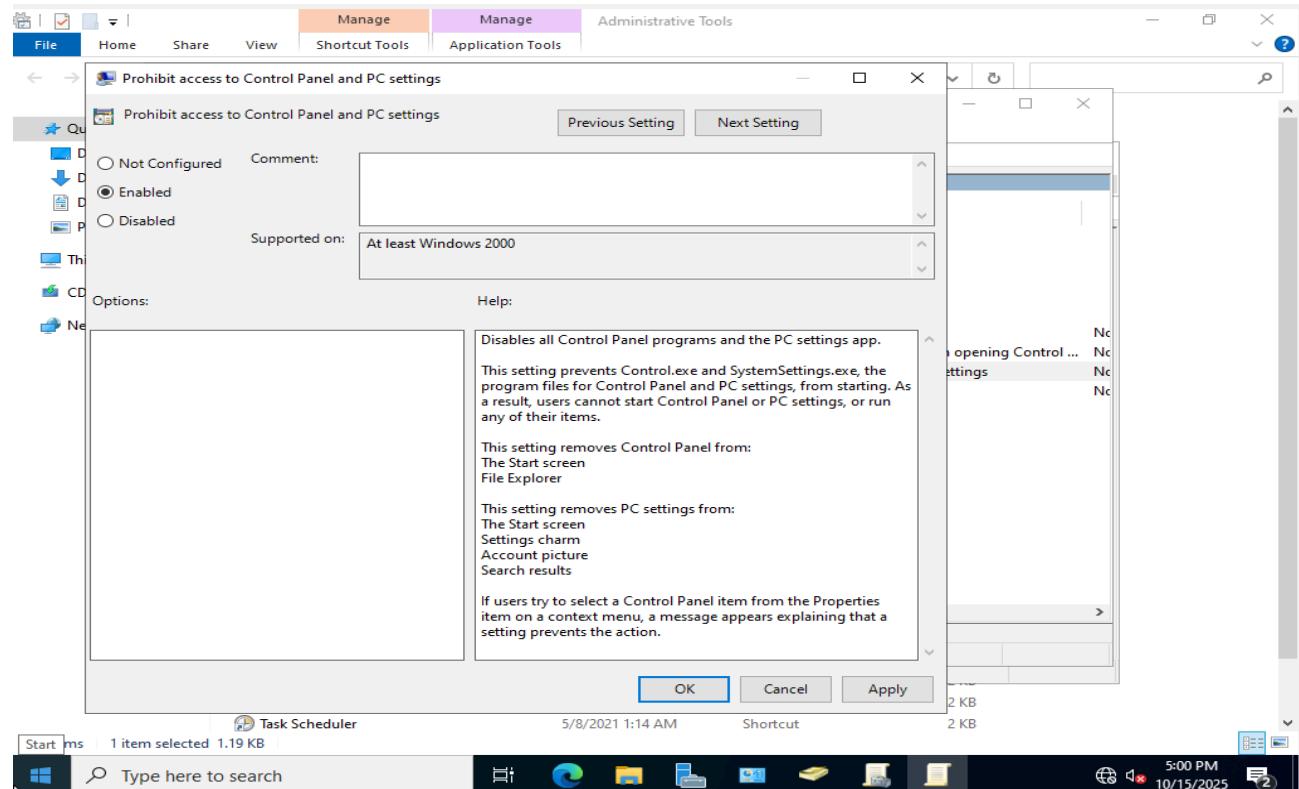
Após a criação de usuários é necessário criar as políticas de grupo que restringem os acessos de cada usuário. Para isso é necessário voltar às ferramentas administrativas, e acessar as opções de Gerenciamento de Políticas de Grupo. A GPO foi nomeada de “pgub”. Nesta política foram configuradas as opções de filtragem, depois foram habilitadas as políticas recomendadas e por fim o relatório da política é exibido ao clicar nas opções.

Figura 91 - Configuração das opções de filtro



Fonte: Elaborado Pelo Grupo

Figura 92 - Políticas recomendadas



Fonte: Elaborado Pelo Grupo

Figura 93 - Relatório da Política

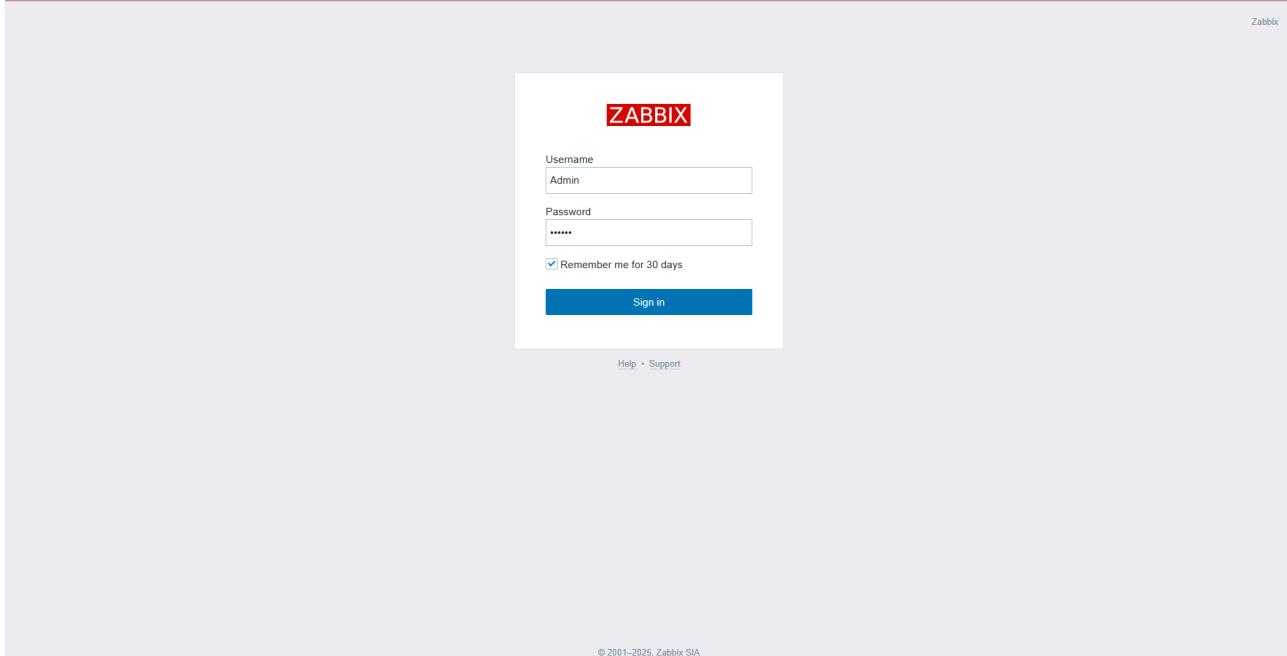
Fonte: Elaborado Pelo Grupo

ETAPA 3- GERÊNCIA E MONITORAMENTO DOS SERVIÇOS DE REDE COM ZABBIX

Durante a etapa 3 foi utilizado o software Zabbix para a obtenção de métricas

através do monitoramento e gerenciamento de dados das máquinas com os serviços configurados na etapa 2. Para isso, foram criadas duas instâncias como servidores Zabbix, um para os hosts no ambiente em nuvem(AWS) e outro localmente(VirtualBox). Uma vez com os servidores corretamente configurados, foi necessário acessar o endpoint “/zabbix” gerado para acessar a interface de monitoramento dos dados. O usuário foi redirecionado para uma tela de login para informar credenciais de acesso de usuário e senha. Por padrão o frontend fornece um usuário administrador com as credenciais de usuário “Admin” e senha “zabbix” com as permissões necessárias. Com isso, o login foi efetuado com as credenciais mencionadas.

Figura 94 - Login no frontend do Zabbix



Fonte: Elaborado Pelo Grupo

Após isso, foi necessário criar os hosts para que o servidor pudesse se conectar às máquinas cliente onde os serviços estavam instalados.

Figura 95 - Criação dos hosts

The screenshot shows the 'Host' configuration page in Zabbix. The 'Host' tab is selected. Key fields include:

- Host name:** AWS_Bruno
- Visible name:** AWS_Bruno
- Templates:** Linux by Zabbix agent
- Host groups:** Linux servers
- Interfaces:** Agent (IP: 13.217.66.82, DNS: AWS_Bruno, Port: 10050, Default)
- Monitored by:** Server
- Description:** (empty)
- Status:** Enabled (checkbox checked)

Buttons at the bottom right include: Update, Clone, Delete, and Cancel.

Fonte: Elaborado Pelo Grupo

Com isso, foi necessário adicionar o hostname de acordo com o que foi anteriormente especificado no arquivo de configuração “zabbix_agentd.conf” do host e a interface de acordo com o ambiente. No caso da imagem acima, foi utilizada a interface “Agent”. Para adicionar um preset das informações que seriam monitoradas foi necessário adicionar o template “Linux by Zabbix agent” baseado nas informações do sistema operacional Linux.

Figura 96 - Hosts Criados e Status de Disponibilidade

Name	Items	Triggers	Graphs	Discovery	Web	Interface	Proxy	Templates	Status	Availability	Agent encryption	Info	Tags
*** AWS_Bruno	Items 75	Triggers 30	Graphs 16	Discovery 3	Web	13.217.66.82:10050		Linux by Zabbix agent	Enabled	ZBX	None	environment:AWS type:client_agent	
*** AWS_Gabriel	Items 98	Triggers 32	Graphs 19	Discovery 5	Web	98.80.113.245:10050		Apache by Zabbix agent, Linux by Zabbix agent	Enabled	ZBX	None	environment:AWS type:client_agent	
*** AWS_Isaac	Items 75	Triggers 30	Graphs 16	Discovery 3	Web	34.197.71.73:10050		Linux by Zabbix agent	Enabled	ZBX	None	environment:AWS type:client_agent	
*** AWS_Matheus	Items 75	Triggers 31	Graphs 16	Discovery 3	Web	98.94.135.116:10050		Linux by Zabbix agent	Enabled	ZBX	None	environment:AWS type:client_agent	
*** Database	Items 147	Triggers 46	Graphs 32	Discovery 4	Web	54.89.195.68:10050		Linux by Zabbix agent, PostgreSQL by Zabbix agent	Enabled	ZBX	None	environment:AWS type:database	
*** Zabbix_Server	Items 154	Triggers 83	Graphs 16	Discovery 6	Web	127.0.0.1:10050		Linux by Zabbix agent, Zabbix server health	Enabled	ZBX	None	type:server	

Displaying 6 of 6 found

Fonte: Elaborado Pelo Grupo

Uma vez que o host foi criado, houve a espera até que a conexão fosse estabelecida, sinalizando “ZBX” com um fundo verde. Caso o fundo ficasse vermelho, significaria que houve um problema de conexão com o host especificado ou que o host se encontra offline no momento.

3.1 Evidências do Monitoramento dos Serviços AWS

As máquinas foram conectadas ao servidor por meio da interface “Agent”. Para isso, primeiramente foram instalados os pacotes de servidor, agente, frontend e banco de dados disponibilizados pelo Zabbix na instância EC2 do servidor. Após isso, o pacote de agente foi instalado em cada instância com os serviços hospedados e os arquivos de configuração foram editados para enxergar o IP do servidor, ajustar o hostname e habilitar a conectividade na porta 10050. Com isso, foram analisados os dados internos do servidor e os resultados gerados pelo gerenciamento dos serviços configurados em nuvem: NFS, servidor web Apache, FTP, DNS e banco de dados PostgreSQL.

3.1.1 Gráficos Gerais

É possível acessar na aba “Hosts”, dentro da seção “Monitoring”, os gráficos gerados de acordo com o template especificado. Para isso, foi necessário acessar “Graphs” de acordo com o agente que será analisado.

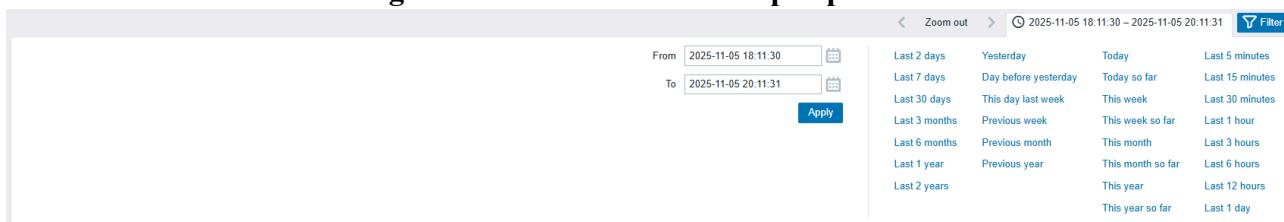
Figura 97 - Informações gerais de monitoramento

Name	Interface	Availability	Tags	Status	Latest data	Problems	Graphs	Dashboards	Web
AWS_Bruno	13.217.66.82:10050	ZBX	class: os environment: AWS target: linux ...	Enabled	Latest data 75	1	Graphs 16	Dashboards 3	Web
AWS_Gabriel	98.80.113.245:10050	ZBX	class: os class: software environment: AWS ...	Enabled	Latest data 98	1	Graphs 19	Dashboards 4	Web
AWS_Isaac	34.197.71.73:10050	ZBX	class: os environment: AWS target: linux ...	Enabled	Latest data 75	1	Graphs 16	Dashboards 3	Web
AWS_Matheus	98.94.135.116:10050	ZBX	class: os environment: AWS target: linux ...	Enabled	Latest data 75	Problems	Graphs 16	Dashboards 3	Web
Database	54.89.195.68:10050	ZBX	class: database class: os environment: AWS ...	Enabled	Latest data 147	Problems	Graphs 32	Dashboards 5	Web
Zabbix_Server	127.0.0.1:10050	ZBX	class: os class: software subclass: logging ...	Enabled	Latest data 154	Problems	Graphs 16	Dashboards 4	Web

Fonte: Elaborado Pelo Grupo

Os dados disponibilizados nos gráficos mostram as estatísticas de acordo com um intervalo de tempo, sendo possível mapear o histórico do comportamento de acordo com um determinado período t.

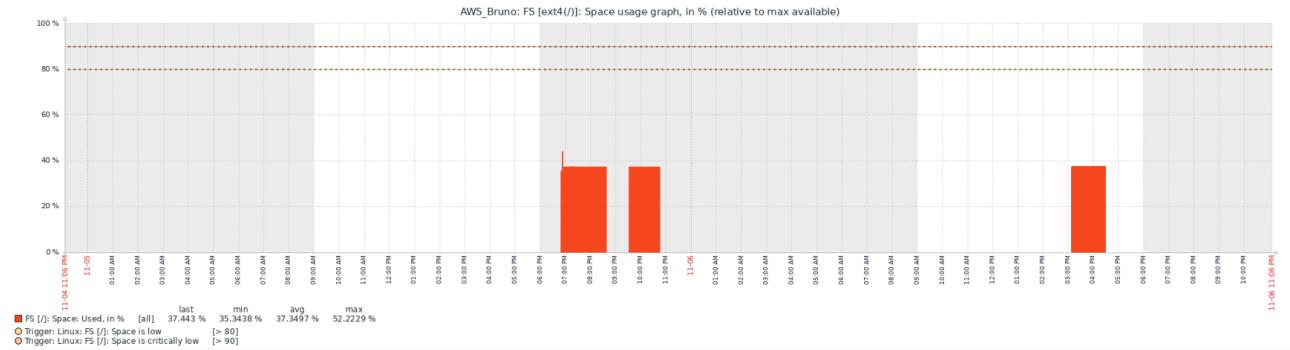
Figura 98 - Filtro de métricas por período



Fonte: Elaborado Pelo Grupo

No caso, os dados do gráfico escolhido foram filtrados para captar informações do dia 05 de novembro de 2025 no horário compreendido entre 18:11 a 20:11, de acordo com o fuso-horário especificado durante a instalação do servidor(UTC-3). O template “Linux by Zabbix agent” abrange no geral muitas métricas de disco, CPU e memória. O host “AWS Bruno” foi utilizado como exemplo:

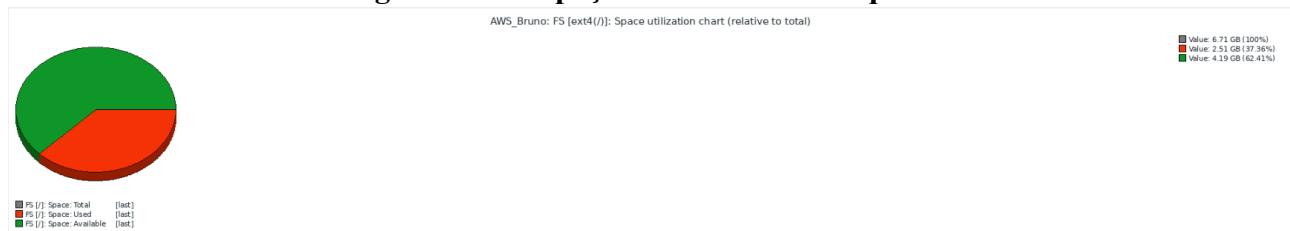
Figura 99 - Espaço em Disco(%)



Fonte: Elaborado Pelo Grupo

A máquina estava utilizando relativamente pouco espaço em disco para executar seus processos, pois como o gráfico indica não houve nenhum disparo dos triggers para sinalizar que há pouco espaço em disco e a porcentagem de uso não ultrapassa nem 40%.

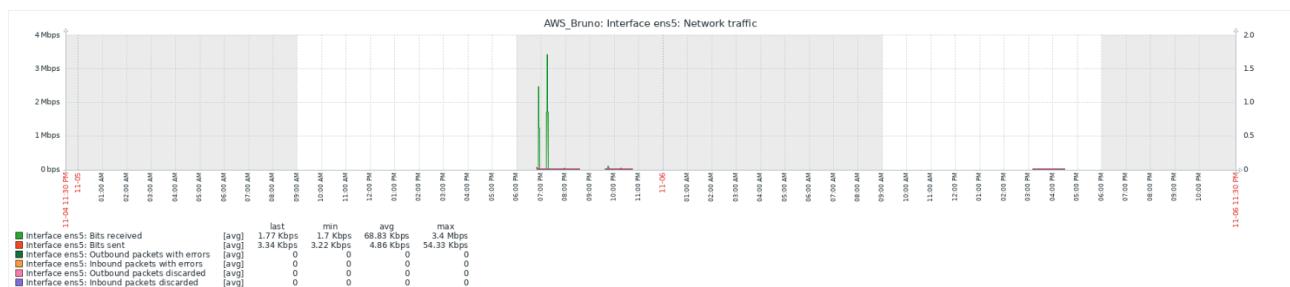
Figura 100 - Espaço do Sistema de Arquivos



Fonte: Elaborado Pelo Grupo

O gráfico indica o espaço disponível no sistema de arquivos(Filesystem) da instância em relação a quanto espaço está sendo utilizado, em bytes. Conforme o gráfico, 2.5GB de 6.7GB estão em uso, ou seja, 64% do sistema de arquivos está ocupado com arquivos e aproximadamente 37% está livre.

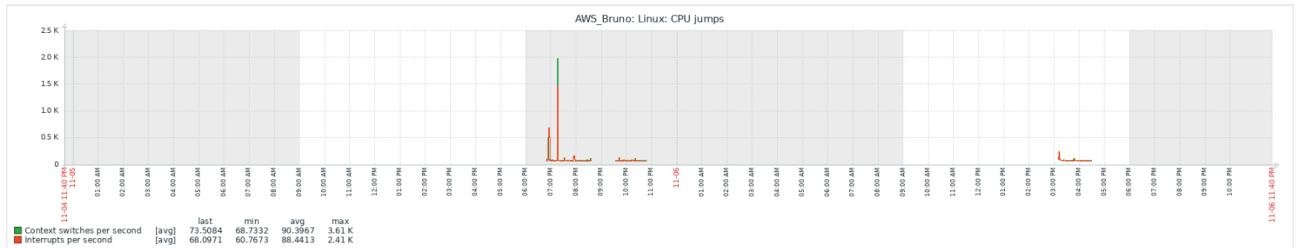
Figura 101 -Tráfego na rede(Bits)



Fonte: Elaborado Pelo Grupo

O tráfego de entrada e saída de dados é continuamente monitorado enquanto a instância EC2 estiver rodando. As informações enviadas e recebidas através do uso do sistema indicavam que havia um elevado fluxo de entrada em relação ao de saída, indicando os bits recebidos e enviados pela instância.

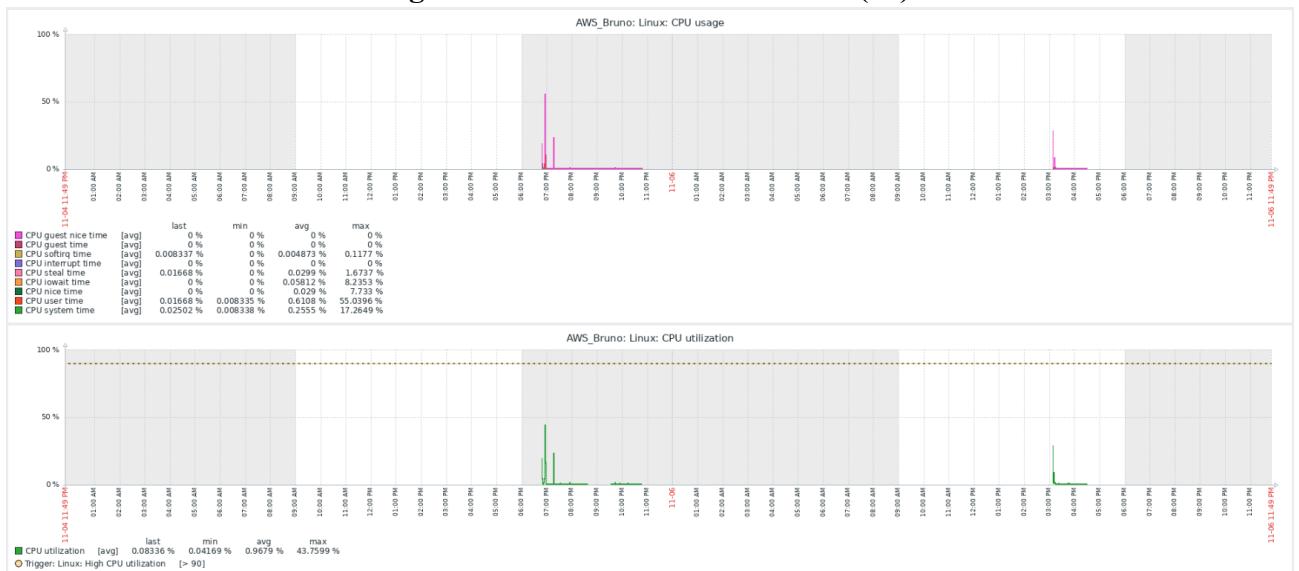
Figura 102 - Variação de uso da CPU



Fonte: Elaborado Pelo Grupo

Em alguns momentos ocorre o uso intensivo da CPU e isso acarreta nos “**CPU jumps**”, momentos em que há uma rápida variação na porcentagem de utilização. Como mostrado no gráfico, em um certo momento a intensidade aumentou pela execução de algum processo mais pesado e se estabilizou logo em seguida, com poucas variações perceptíveis.

Figura 103 - Média de Uso da CPU(%)

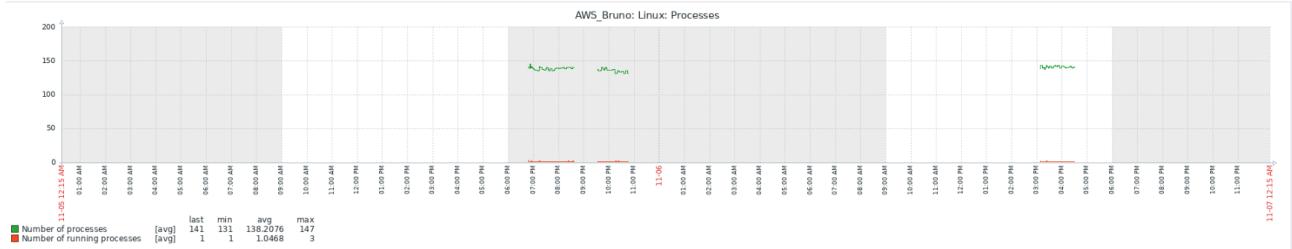


Fonte: Elaborado Pelo Grupo

Nota-se que os CPU jumps impactam diretamente nos resultados de uso da máquina, pois quando há uma maior variação na atividade da máquina consequentemente a utilização de sua CPU aumenta drasticamente. Ao mesmo tempo, pode-se verificar que a comunicação da máquina com o servidor físico está correndo de forma tranquila, indicado pelo gráfico “**CPU usage**”. Estão mapeados timestamps(registros temporais) em que a CPU está em modo “steal time” ou “tempo de roubo”, tipicamente quando a comunicação de transferência para

um servidor está ocorrendo, assim como “user time”, quando está sendo ativamente usada.

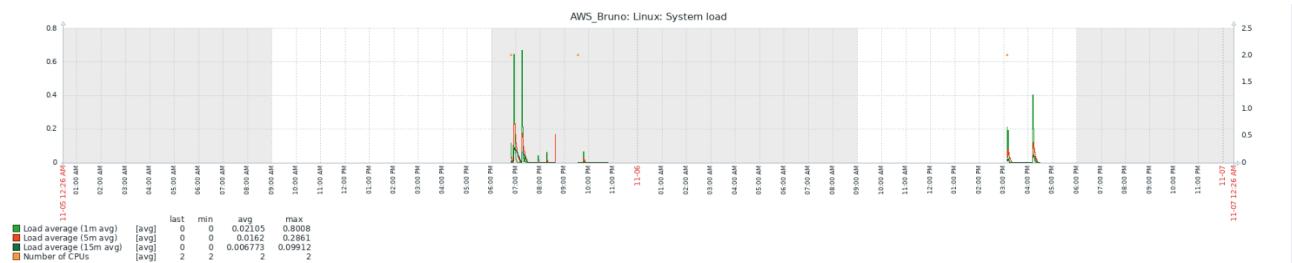
Figura 104 - Processos



Fonte: Elaborado Pelo Grupo

A CPU estava rodando de forma otimizada e sem saturação de processos. O número de processos que efetivamente rodavam era bem inferior ao total, o que é uma conduta normal de sistemas Linux, pois no boot(inicialização) os processos já se encontram armazenados no sistema para serem executados quando necessário.

Figura 105 - Carregamento do sistema



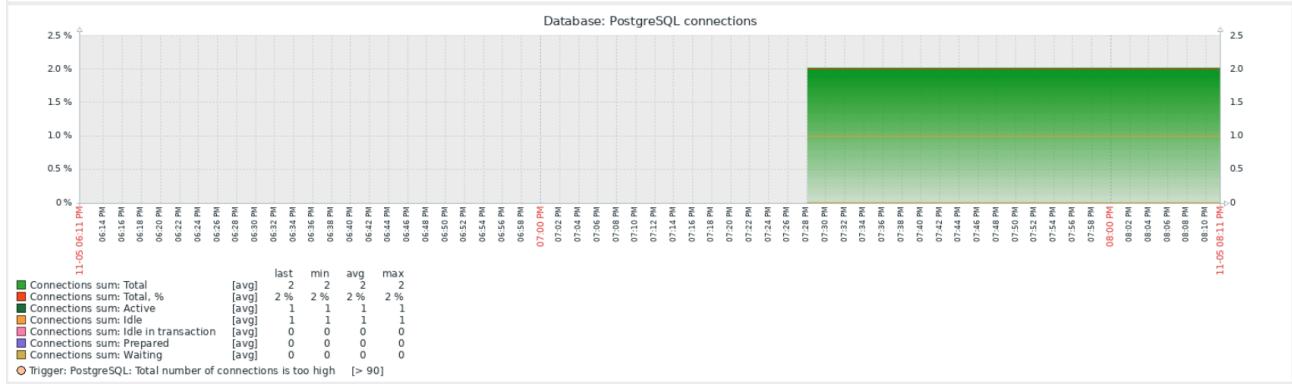
Fonte: Elaborado Pelo Grupo

O gráfico demonstra o tempo de espera médio para que os processos da instância sejam executados. Como isso depende do número de núcleos(core) da máquina, no caso do tipo t2.micro, que possui dois núcleos, todo o gerenciamento para o carregamento dos processos será baseado nisso. É possível perceber que o tempo de carregamento apresenta algumas instabilidades devido a processamentos mais pesados.

3.1.1.1 Banco de Dados PostgreSQL

Para o monitoramento das informações transmitidas pelo banco de dados foi utilizado o template “**PostgreSQL By Zabbix agent**” e terá que ser feita a configuração especificada na descrição ou no repositório git de acordo com a versão do Zabbix. No caso, a branch “**release/7.4**” foi a utilizada.

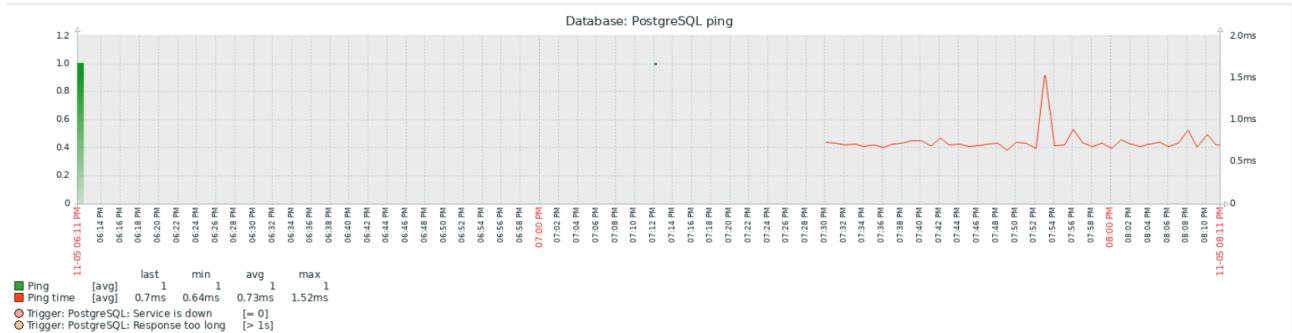
Figura 106 - Conexões



Fonte: Elaborado Pelo Grupo

As conexões feitas por usuários no banco de dados estavam sendo captadas, indicando a comunicação com o agente PostgreSQL. Foi possível verificar que dentro do intervalo de tempo especificado houveram mais requisições no período final, o que indica que anteriormente o banco estava sendo pouco utilizado ou inativo.

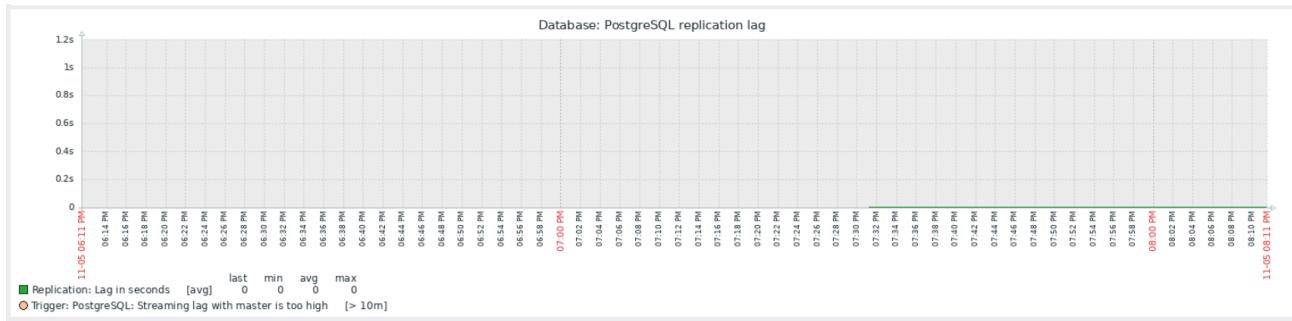
Figura 107- Ping



Fonte: Elaborado Pelo Grupo

A métrica de ping evidenciava que a latência do banco esteve mais elevada especificamente na metade do tempo em que o banco esteve em atividade, chegando a 1.5ms(milissegundos) de delay, e durante o restante do tempo permaneceu estável, com uma média de 0.5 a 1 ms de velocidade de resposta. Possíveis causas para o pico de atividade analisado no timestamp, quando a resposta chega a uma latência mais alta, podem ser o maior processamento de informações ou principalmente a tentativa de transmitir dados via FTP.

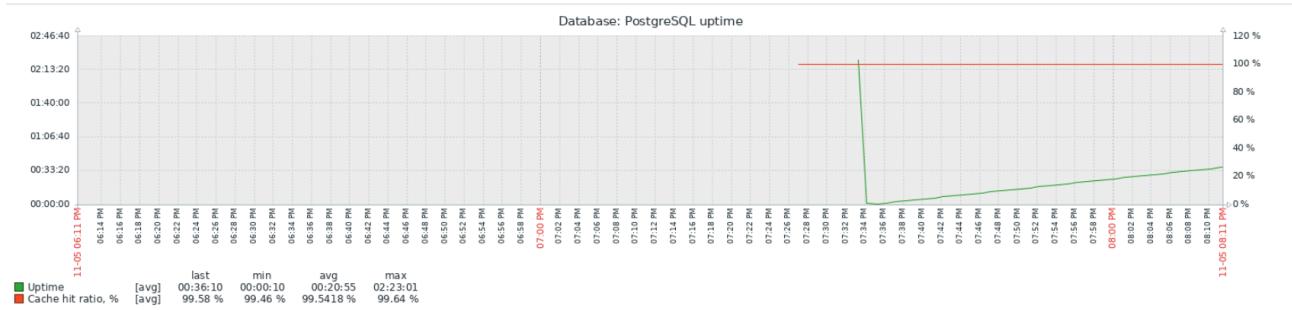
Figura 108 - Status de Lag



Fonte: Elaborado Pelo Grupo

As replicações de lag do banco de dados mostraram que não houveram falhas significativas durante o processamento do banco de dados e que as respostas às requisições se mantiveram estáveis. Isso indica o bom funcionamento do banco.

Figura 109 - Uptime

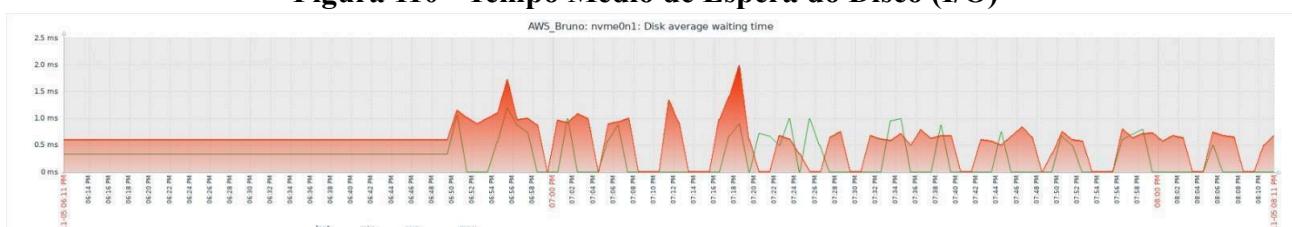


Fonte: Elaborado Pelo Grupo

A figura indica os horários em que o banco esteve ligado e o status da utilização do armazenamento em cache. É possível ver em dado momento que a atividade do banco chega a 0% e gradualmente começa a aumentar novamente, indicando que a instância foi reiniciada.

3.1.1.2 Monitoramento do NFS

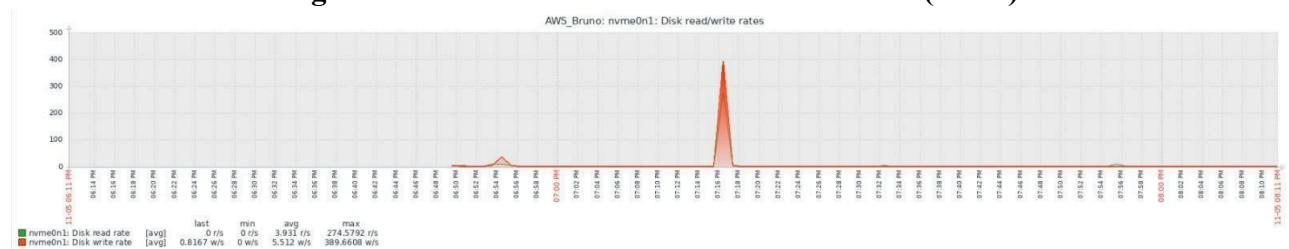
Figura 110 - Tempo Médio de Espera do Disco (I/O)



Com a análise de performance do disco, a Figura 3.1.1.2.1 detalha o tempo médio de espera (latência) para as requisições de I/O (entrada/saída) do servidor. O gráfico compara o tempo de espera para requisições de escrita “**w_await**” com o tempo de espera para leitura “**r_await**”.

Nota-se que a latência de leitura se manteve consistentemente baixa, com uma média de 0.27ms. Em contrapartida, a latência de escrita apresentou uma volatilidade muito maior, especialmente no período entre 18:52 e 20:11, onde ocorreram diversos picos. O pico máximo de espera de escrita chegou a quase 2.0ms (por volta das 19:22), indicando que as operações de gravação de dados no servidor NFS foram a principal fonte de latência de disco durante os momentos de maior utilização.

Figura 111 - Taxas de Leitura/Escrita do Disco (IOPS)

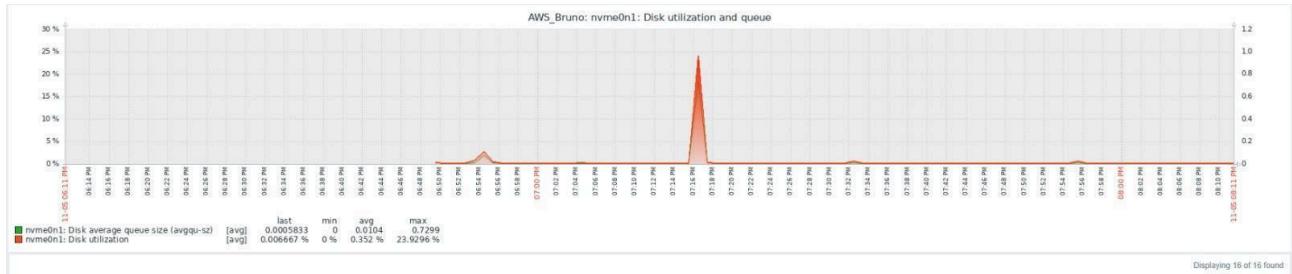


Fonte: Elaborado Pelo Grupo

A Figura 3.1.1.2.2 detalha o número de operações de entrada e saída(I/O) por segundo do disco, separando as taxas de leitura “**Disk read rate**” das taxas de escrita “**Disk write rate**”.

O gráfico reforça o padrão de uso esporádico do servidor NFS. Durante a maior parte do período analisado, o disco permaneceu em inatividade. Contudo, um evento de alta intensidade é claramente visível por volta das 19:20. Nesse curto intervalo, a taxa de escrita atingiu um pico máximo de 389 w/s (operações de escrita por segundo), enquanto a taxa de leitura captou um pico simultâneo de 214 r/s (operações de leitura por segundo).

Este pico é a causa direta do aumento de latência de escrita visto na Figura 100. Isso indica que uma carga de trabalho súbita e intensa exigiu um alto número de operações de gravação do disco, impactando momentaneamente seu tempo de resposta.

Figura 112 - Utilização e Fila do Disco

Fonte: Elaborado Pelo Grupo

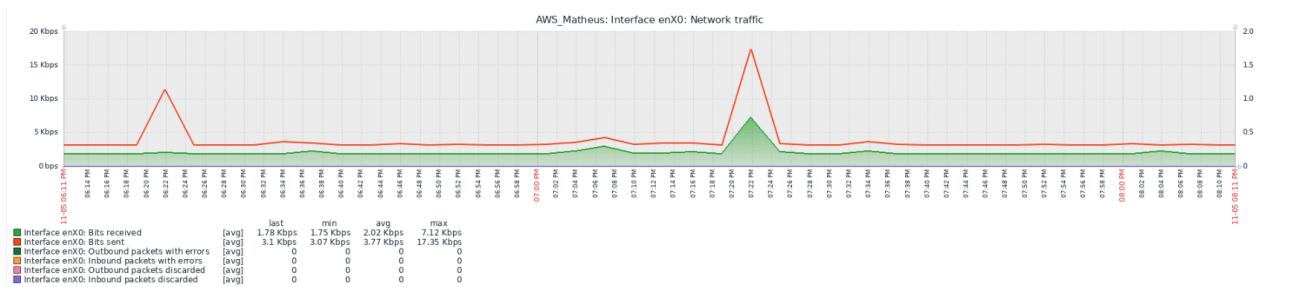
Este gráfico monitorava dois indicadores críticos da saúde do disco: a utilização “Disk Read Utilization” e o tamanho médio da fila “Disk Average Queue Size”. A utilização mede a porcentagem de tempo que o disco esteve ocupado, enquanto a fila indica quantas requisições estavam aguardando para serem processadas.

Como esperado, os dados acompanhavam perfeitamente os picos de atividade vistos nas Figuras 3.1.1.2.1 e 3.1.1.2.2. No momento de maior intensidade (por volta das 19:20), a utilização do disco atingiu seu máximo de 23.9%. Isso mostra que, mesmo durante a maior carga de trabalho registrada, o disco ainda possuía mais de 75% de capacidade livre, indicando que não está sobrecarregado.

Simultaneamente, a fila de requisições (linha verde, quase imperceptível na base do gráfico) permaneceu próxima de zero (média de 0.01), com um pico máximo de apenas 0.7. Isso é um sinal excelente, pois indica que as requisições não estão "empilhando" e sendo rapidamente processadas pelo disco.

3.1.1.3 Monitoramento do FTP

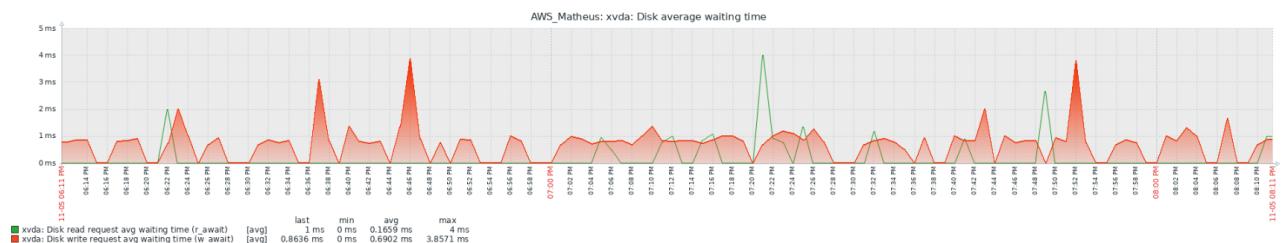
No caso do serviço FTP foram analisadas as métricas de entrada e saída de dados nas redes para captar atividades de compartilhamento de arquivos entre um cliente FTP e o servidor através das portas estabelecidas pelo protocolo, tal como os dados de leitura de disco para captar atividades de criação de pastas no servidor.

Figura 113 - Informações da interface do tráfego de rede

Fonte: Elaborado Pelo Grupo

Foi possível analisar oscilações de bits enviados e recebidos através da interface de rede do servidor FTP. O gráfico indicava picos de tráfego de rede no início e no meio do período analisado. Os picos se justificam pelo fato de que nesses horários usuários estariam fazendo requisições para se conectar ao servidor FTP.

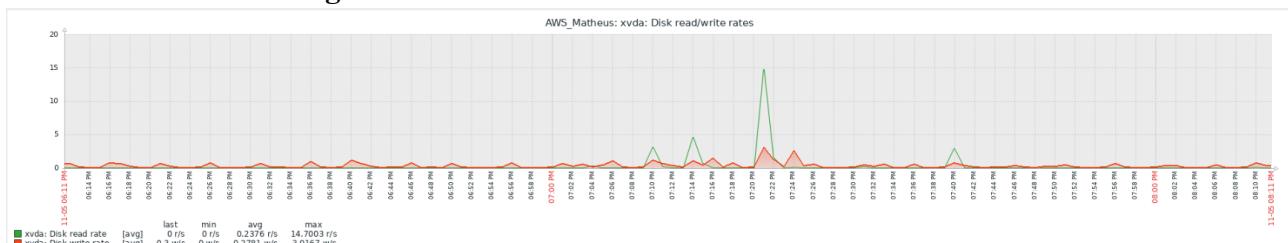
Figura 114 - Latência de operações no disco



Fonte: Elaborado Pelo Grupo

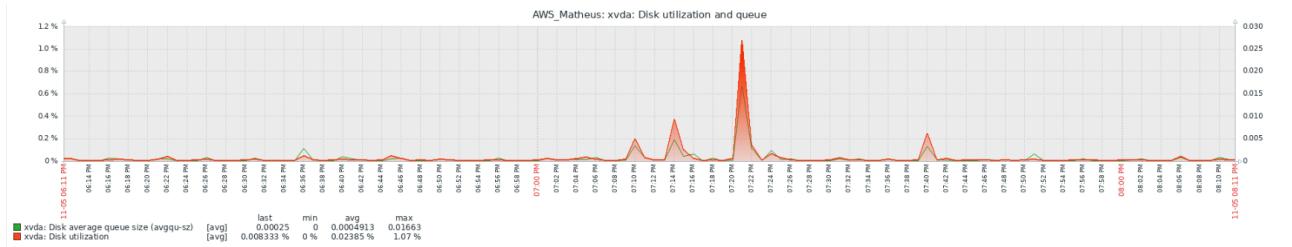
Conforme a figura, a elevada variação no tempo de leitura e escrita durante o período analisado evidenciava a variação da latência entre operações de escrita de arquivos no volume do servidor FTP e requisições de leitura pela listagem dos arquivos remotos para um host cliente. Houve um momento específico em que o tempo de leitura chegou a 4 ms, o que denota uma falha de conexão com o servidor remoto para ler os arquivos neste momento, principalmente quando se tentou efetuar a conexão do banco de dados via FTP para o armazenamento de arquivos disponibilizados pelo servidor.

Figura 115 - Velocidade de leitura e escrita de disco



Fonte: Elaborado Pelo Grupo

O gráfico evidencia que as requisições de um cliente para a listagem dos dados disponibilizados pelo servidor FTP foram, no geral, mais abundantes do que as operações de escrita. Foi possível analisar que em dado momento houve um pico de leitura, provavelmente por logs(registros) de erro sequenciais causados por falhas à tentativa de listar os diretórios do servidor.

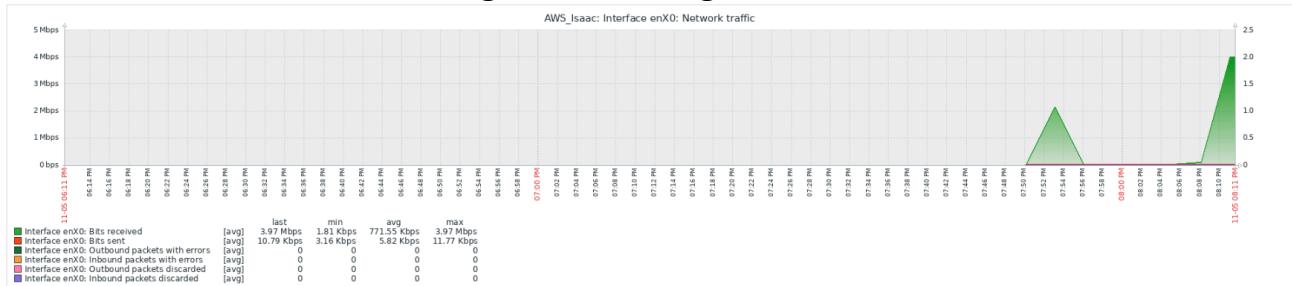
Figura 116 - Utilização e processos em fila do disco

Fonte: Elaborado Pelo Grupo

A figura mostra que houve um pico no processamento em disco em determinado período mas não o suficiente para causar um bottleneck dos processos em fila(aguardando para execução). Isso geralmente ocorre em situações de uso mais intensivas, o que condiz com a tentativa de conexão do servidor FTP com o banco de dados. No geral, o gráfico aponta para uma utilização mais leve do disco, evidenciando que não há nenhuma falha significativa com relação ao disco e que o comportamento está dentro do ideal.

3.1.1.4 Monitoramento DNS

As principais métricas necessárias para extrair o funcionamento do servidor DNS serão a análise e o tráfego na interface de rede, identificando os status de tráfego na rede através do status das zonas e nome de domínio configurados.

Figura 117 - Tráfego de rede

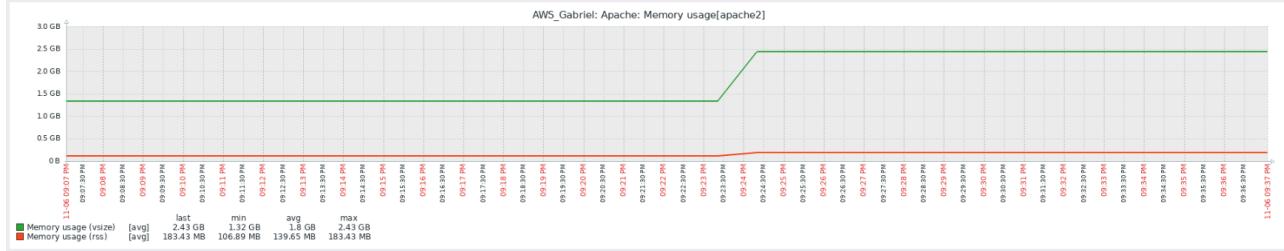
Fonte: Elaborado Pelo Grupo

Foi possível perceber um pico no tráfego de entrada atividade de rede ao executar lookups durante o tempo em que a instância permaneceu ligada. Até o final do período filtrado não havia dados, pois a máquina estava desligada. Com isso, o tráfego de dados e disponibilidade do servidor está ocorrendo dentro do esperado conforme as respostas recebidas pelo comando “dig”.

3.1.1.5 Monitoramento Web

As métricas consideradas mais importantes para analisar o desempenho do servidor web Apache foram relacionadas ao comportamento das requisições HTTP aos endpoints criados e ao gerenciamento do status das respostas em formato html recebidas. Para exibir os dados do servidor Apache foi adicionado o template “**Apache by Zabbix agent**”, responsável por capturar as métricas disponibilizadas pelos arquivos de configuração do pacote apache2.

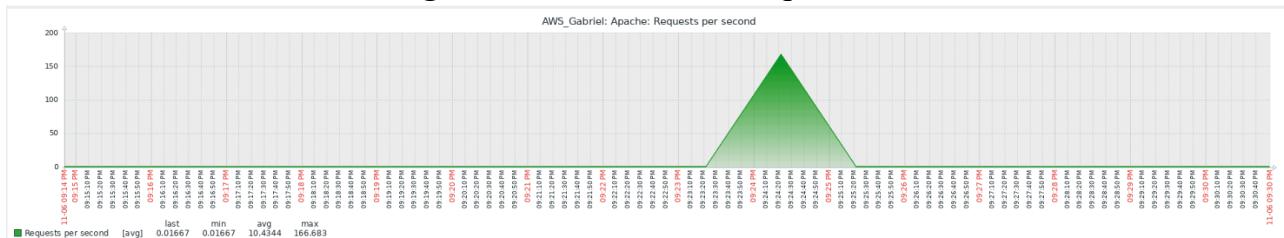
Figura 118 - Uso de Memória pelo Apache



Fonte: Elaborado Pelo Grupo

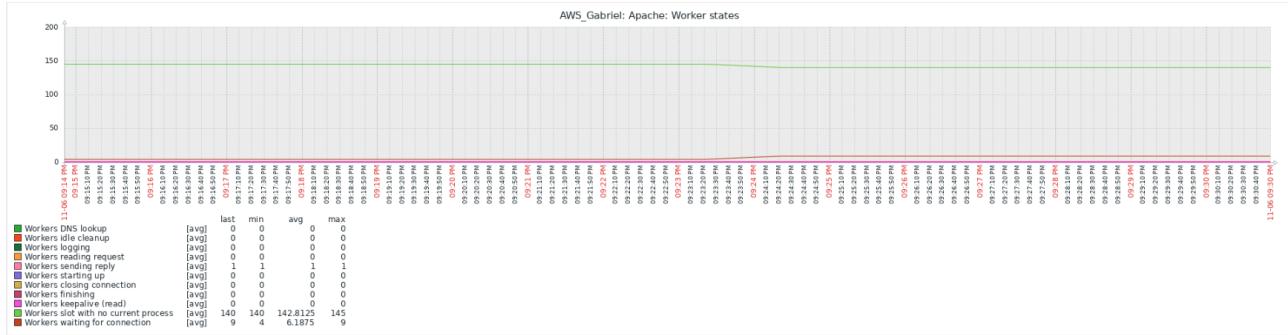
O gráfico demonstra que o uso de memória do Apache se manteve estável e houve apenas uma pequena oscilação no meio do período, demonstrando o leve aumento de processamento de requisições no uso de dados web durante esse dado intervalo de tempo. Essa oscilação ocorreu por ter sido enviado o comando “**ab -n 10000 -c 100 [servidor_web]**”, em que “[servidor_web]” é o endereço web Apache. Isso dispara várias requisições HTTP automaticamente e consequentemente força o uso da memória RAM.

Figura 119 - Demanda de requisições

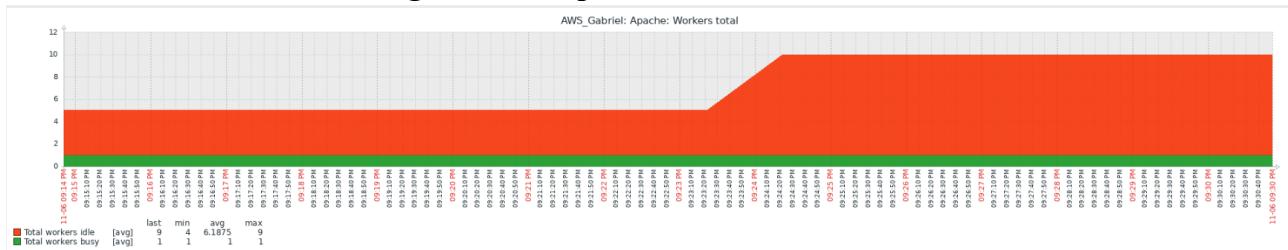


Fonte: Elaborado Pelo Grupo

O fator mencionado anteriormente impactou diretamente no gráfico de requisições por segundo, pois é possível ver que no geral a média de requisições se mantém estável até um timestamp em que há um pico abrupto do consumo do servidor, chegando a uma média de 166 requisições por segundo.

Figura 120 - Apache: Estados dos Workers**Fonte:** Elaborado Pelo Grupo

O Apache possui um sistema multithreading capaz de processar requisições em larga escala através da divisão de tarefas em várias threads, ou núcleos, diferentes. Para isso, cada serviço worker se encarrega de separar esses processos em uma thread de execução fora do núcleo de processamento principal, evitando o travamento de interfaces. No contexto apresentado, é possível perceber que todos os workers estão inativos ou esperando por conexões. Isso sinaliza o poder de processamento de um sistema multi-thread, pois mesmo um volume razoável de dados não demanda poder de processamento o suficiente para ativar os jobs nos workers.

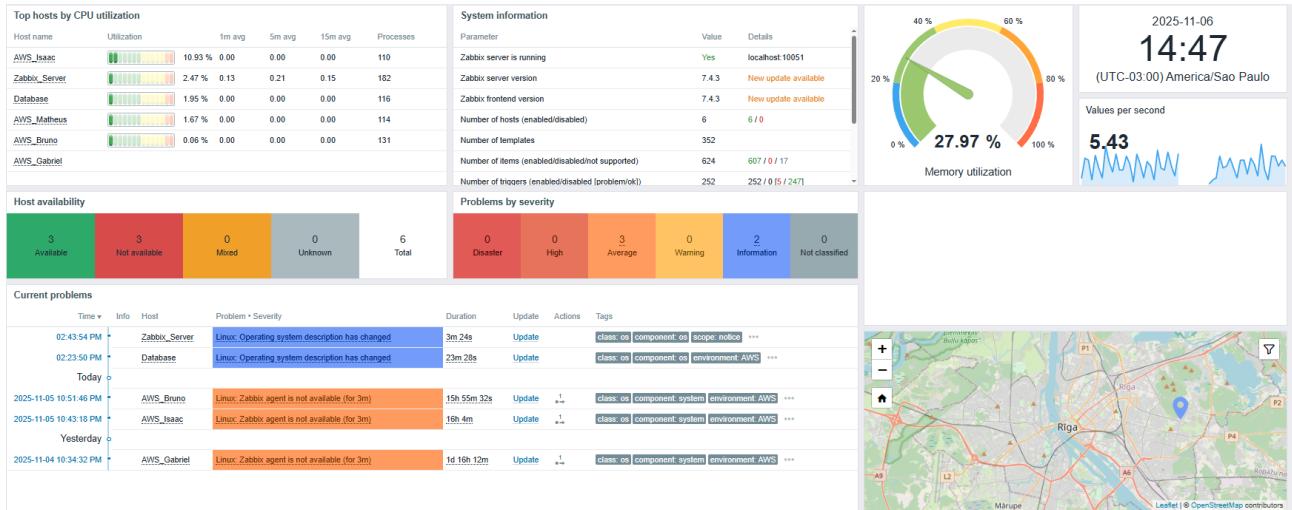
Figura 121 - Apache: Total de Workers**Fonte:** Elaborado Pelo Grupo

O gráfico evidencia ainda mais a pouca utilização dos workers para executar processos via multi-thread. Entretanto, aqui pode-se notar que conforme o consumo de memória pela thread principal o Apache cria mais workers, mesmo que sem estarem ocupados. Isso demonstra a eficiência do sistema para o planejamento em execução multithread antes mesmo que ela seja realmente necessária.

3.1.2 Métricas Dashboard

Para acessar os templates de dashboards, é necessário acessar a aba “Dashboards” e “All Dashboards”. Nativamente o Zabbix possui um template de dashboards chamado “Global view”, que possui dados genéricos sobre as informações de monitoramento das máquinas e do servidor Zabbix.

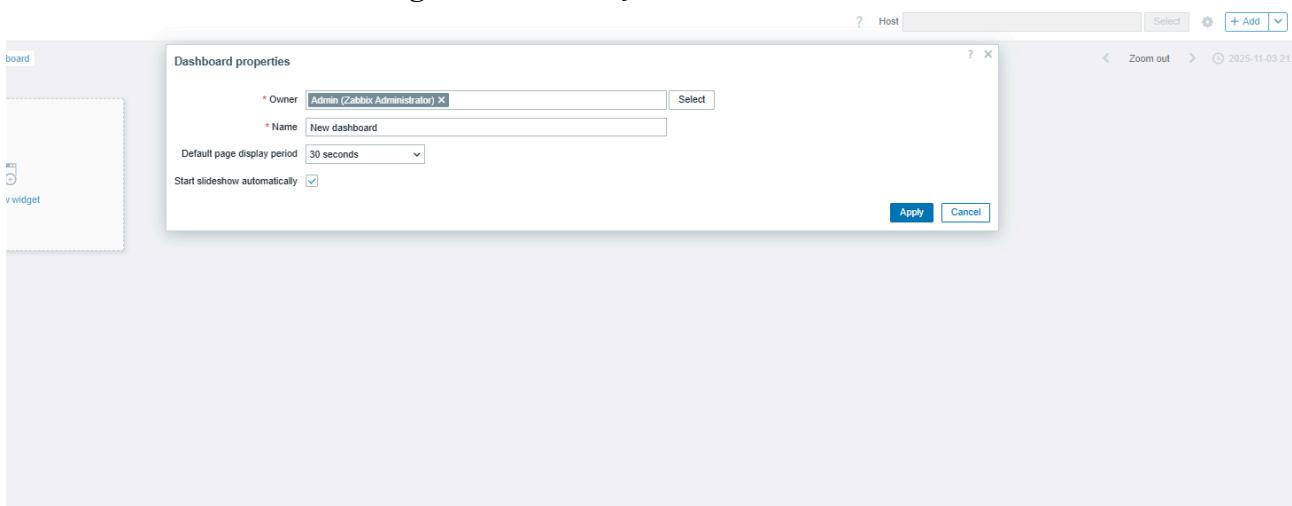
Figura 122 - Informações de Global view



Fonte: Elaborado Pelo Grupo

Esse dashboard mapeia uma ideia geral dos dados dos hosts e de desempenho do servidor, entretanto não há nenhum dado específico sobre os hosts a serem monitorados. Portanto, será necessário criar um novo template de dashboard. Basta voltar para a guia “**All Dashboards**” e pressionar “**Create dashboard**”. Com isso, uma janela irá aparecer.

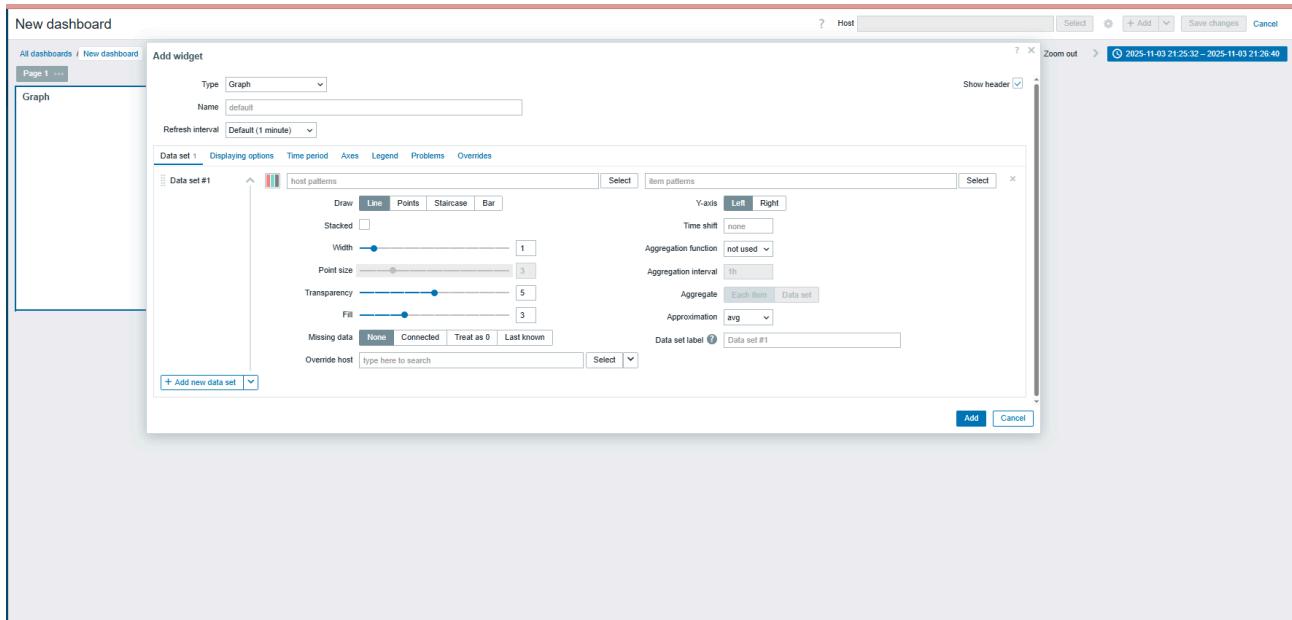
Figura 123 - Criação de um dashboard



Fonte: Elaborado Pelo Grupo

Será necessário escolher o administrador Zabbix como dono e clicar em “**Apply**”. Com isso será gerada uma página e poderão ser adicionadas cada informação desejada ao selecionar “**Add widget**”.

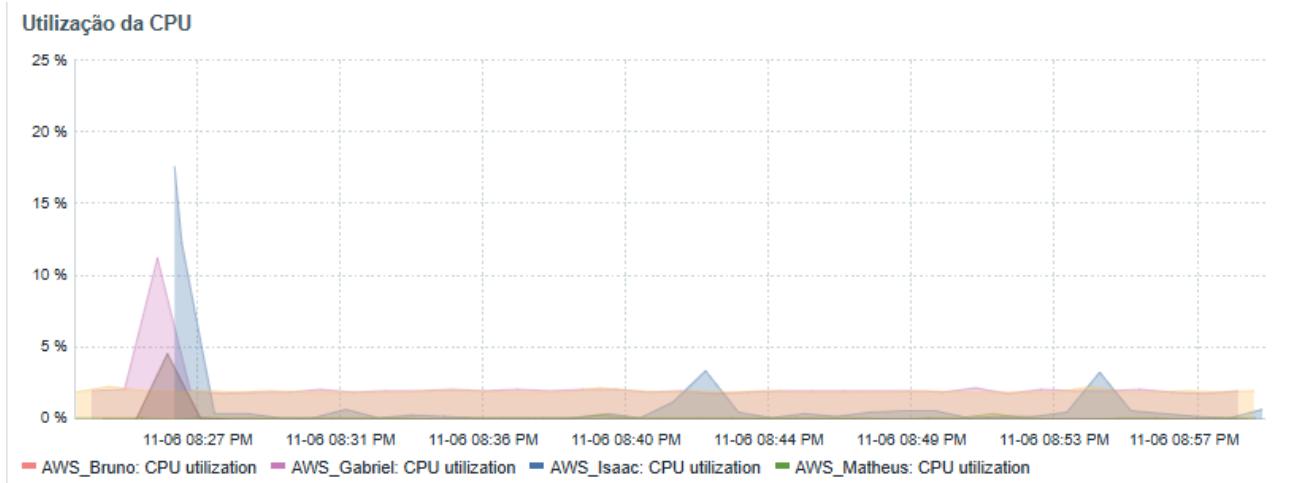
Figura 124 - Criação de uma estatística



Fonte: Elaborado Pelo Grupo

Para gerenciar o uso de CPU e memória de cada serviço com exceção do banco de dados, foram criados dois gráficos principais:

Figura 125 - Monitoramento da CPU no dashboard



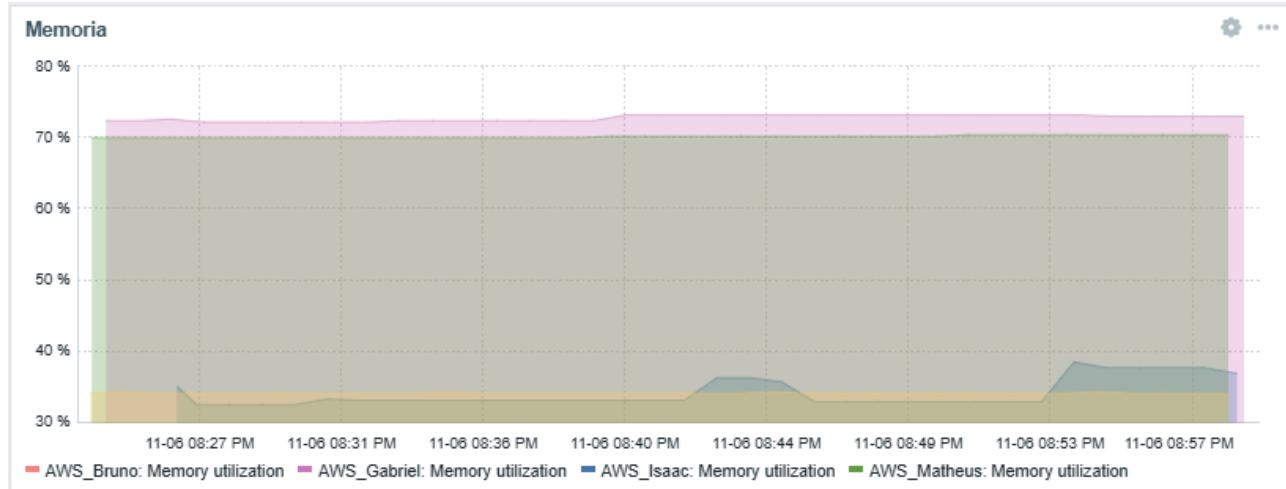
Fonte: Elaborado Pelo Grupo

É possível analisar no gráfico acima o consumo geral da CPU dos agentes Zabbix, indicado para cada instância conforme a legenda. Percebe-se que, em média, o host com o **DNS(AWS_Isaac)** exigiu mais da CPU em comparação aos outros agentes, chegando a mais do que 15% de utilização. Em contraste, como o serviço **FTP(AWS_Matheus)** possui um método de operação muito simples, visto que apenas transfere arquivos. Como o DNS trabalha com queries e lookups, funcionalidades mais complexas, é normal verificar essa disparidade.

O servidor **Apache(AWS_Gabriel)** possui um pico de atividade no começo e depois pouca ou nenhuma atividade. A explicação para isso seria a execução de um teste de stress

específico em que várias requisições HTTP são enviadas de uma vez só, demandando um processamento repentinamente maior da CPU. Por último, tem-se o **servidor NFS(AWS_Bruno)**, que possui uma alta estabilidade. Essa estabilidade mostra como o gerenciamento de disco tenta manter um balanceamento de carga da CPU e o controle eficiente do compartilhamento de arquivos entre o servidor e um cliente que solicita o acesso a esses arquivos.

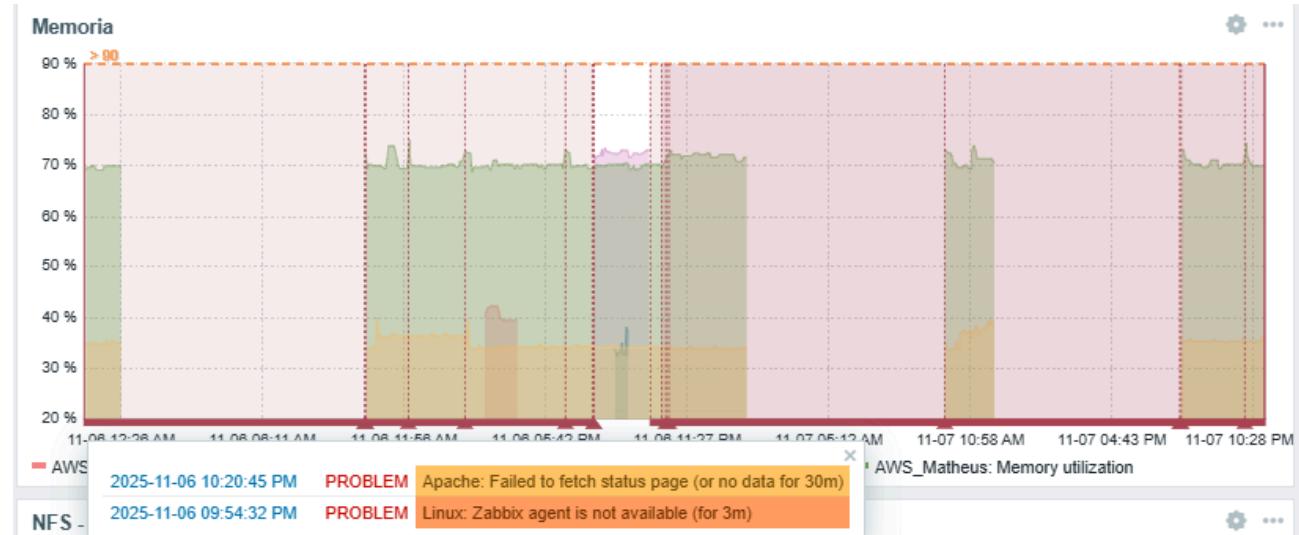
Figura 126 - Monitoramento de memória no dashboard



Fonte: Elaborado Pelo Grupo

Conforme presente nos gráficos, as duas máquinas mais relevantes no cenário de utilização da memória RAM são o servidor web e o ftp. Durante a atividade dessas máquinas, a memória RAM foi constantemente utilizada em aproximadamente 70% de sua capacidade, indicando possíveis processos pesados rodando sem o conhecimento do usuário ou que tenham sido instalados por engano. Como o Apache é leve e a transferência de arquivos FTP normalmente exige o mínimo, é um cenário um tanto quanto incomum. Para encontrar possíveis problemas encontrados durante o período, foi utilizada a opção de mostrar problemas.

Figura 127 - Debug de erros



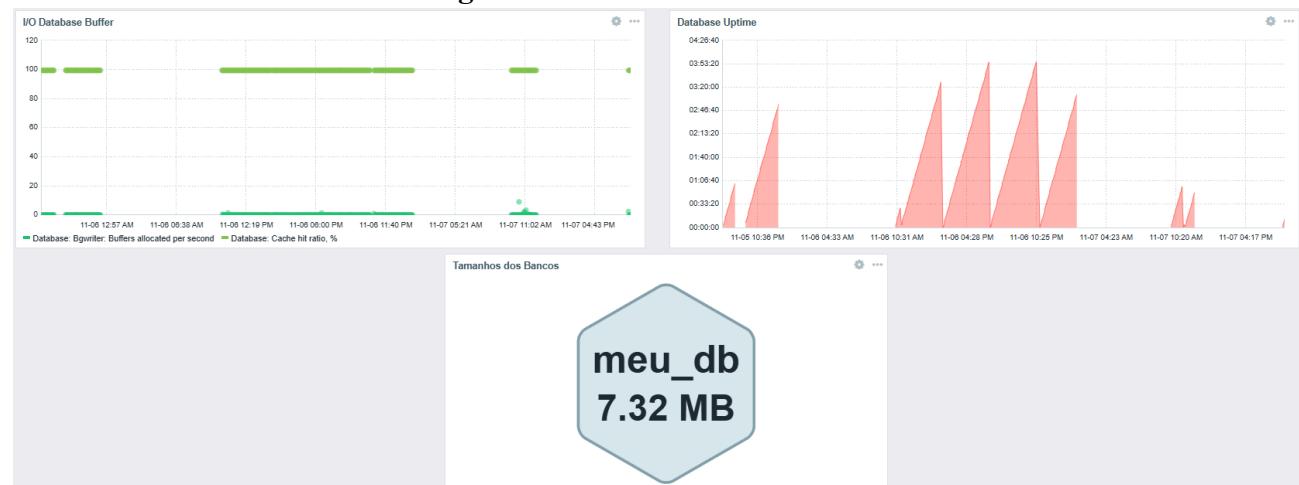
Fonte: Elaborado Pelo Grupo

Mesmo com os problemas encontrados, não há um motivo evidente para ter esse impacto nas instâncias, pois estão apenas relacionados à disponibilidade. Caso deixar as instâncias ligadas por muito tempo aumente muito o consumo da memória, então seria a razão mais plausível, visto que vários testes específicos foram feitos em ambos os serviços.

3.1.2.1 Monitoramento do Banco de Dados PostgreSQL

Foram utilizadas métricas específicas para analisar os dados transmitidos pelo agente PostgreSQL e analisar o comportamento padrão do banco de dados em um dado intervalo de tempo, de acordo com os presets do template “PostgreSQL By Zabbix agent”. As principais métricas analisadas foram: streams de escrita e leitura nos buffers, tempo de disponibilidade e tamanho do banco criado no terminal da instância “**meu_db**”.

Figura 128 - Métricas analisadas

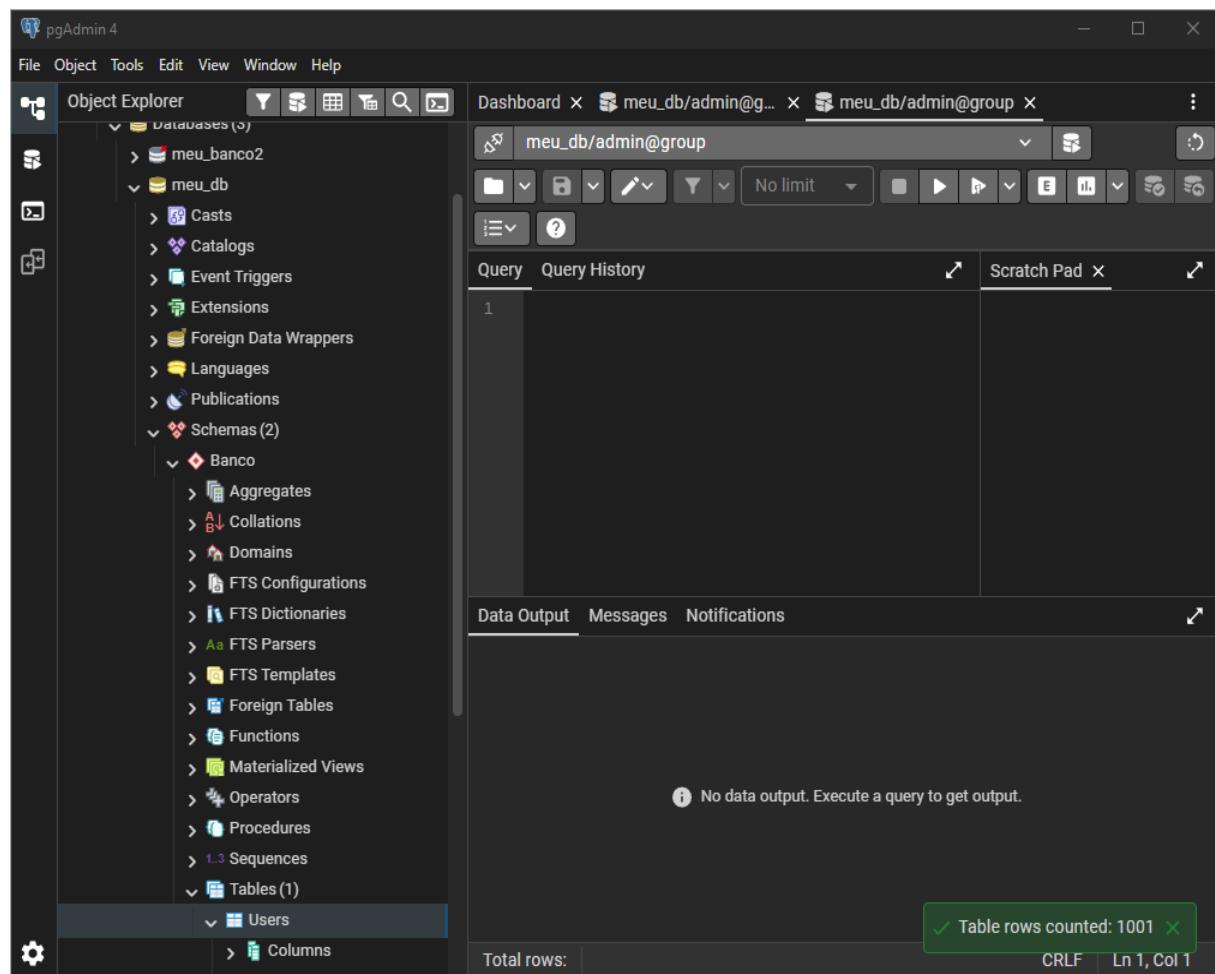


Fonte: Elaborado Pelo Grupo

Abaixo tem-se as especificações sobre o que cada métrica citada monitora:

- **I/O Database Buffer:** Muito importante para analisar a performance do banco de dados, uma vez que demonstra uma relação especial: Quantidade de buffers alocados por segundo em relação ao uso do sistema de cache. Os buffers alocados por segundo indicam a frequência com que dados foram registrados sem a utilização de cache, e caso esse número seja elevado pode acarretar em problemas porque o banco fica sobrecarregado com processos de escritas e leitura ao invés de fornecer os dados direto do cache onde os buffers já estariam armazenados. Como o gráfico demonstra, a porcentagem de cache hit ratio é elevada, que representa as vezes em que houve uma operação “hit” para fazer uma requisição ao cache. Portanto, o uso do cache é elevado e consequentemente os streams de leitura e escrita(I/O) de dados sem a utilização do armazenamento em cache são raros. Conclui-se assim que a configuração do serviço está performática e otimizada.
- **Database Uptime:** Demonstra o tempo em que o banco de dados ficou online para uso. Especificamente mapeia o tempo de duração em que o serviço do postgres esteve disponível para uso e quando esteve indisponível. O padrão percebido é causado pela reinicialização da instância EC2, como é possível ver ao inicializar uma máquina o tempo de disponibilidade começa a aumentar gradualmente, até que seja desligada novamente. No caso, a maioria dos desligamentos ocorreu automaticamente pelo tempo de expiração de 4 horas estabelecido pela AWS. Isso explica os picos do gráfico apontando em média para 03:53:20 como maior valor de uptime atingido.
- **Tamanho dos Bancos:** O valor de 7.32MB indica o tamanho do volume de dados específica do banco de dados “**meu_db**”, criado pelo terminal. Como o template não permite visualizar outro banco além do especificado nas configurações de macro, é necessário alterar o valor para visualizar informações de bancos distintos. Para o teste, foi criada uma tabela nos schemas do banco “**meu_db**” com um campo de texto de inseridas 1000 linhas. Após isso, o tamanho aumentou para 7.44MB.

Figura 129 - Contagem da quantidade de linhas na tabela Users



Fonte: Elaborado Pelo Grupo

Figura 130 - Tamanho do banco após alterações

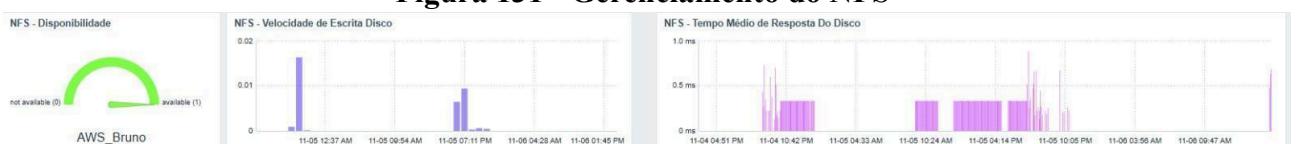


Fonte: Elaborado Pelo Grupo

3.1.2.2 Monitoramento do Servidor NFS

Para o monitoramento das informações e do desempenho do servidor de arquivos NFS (Network File System), foi implementada a coleta de métricas utilizando o Zabbix. O ambiente consiste em um servidor NFS rodando em uma máquina virtual (VM) na AWS, com o sistema operacional Ubuntu. Para a coleta dos dados, foi utilizado o template “**Linux by Zabbix agent**” e a configuração segue as especificações do agente Zabbix. A versão da plataforma Zabbix utilizada para este monitoramento é a 7.4.

Figura 131 - Gerenciamento do NFS



Fonte: Elaborado Pelo Grupo

A figura apresenta um conjunto de métricas essenciais de desempenho do servidor NFS, hospedado na instância “**AWS_Bruno**”. O primeiro indicador, “**NFS - Disponibilidade**”, utiliza um medidor que confirma o status 'available' (1), assegurando que o serviço está online e respondendo.

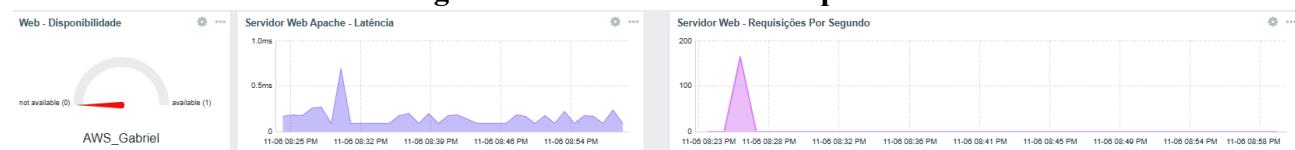
Ao lado, os gráficos analisam a performance do disco: o de “**Velocidade de Escrita Disco**” evidencia um padrão de uso esporádico, com picos de atividade de gravação (como o ocorrido em 05/11 por volta das 00:07) e longos períodos de inatividade. Complementando, o gráfico de “**Tempo Médio de Resposta Do Disco**” mostra que, mesmo durante os períodos de atividade, a latência do disco se manteve saudável e baixa, estabilizada em torno de 0.3ms e com

picos isolados que não ultrapassam 1.0ms. Em conjunto, essas métricas indicam um serviço disponível e com boa performance de armazenamento para a carga de trabalho observada.

3.1.2.3 Monitoramento do Servidor Web

Para o monitoramento do servidor web as principais métricas no dashboard foram o tempo de resposta(latência) dos endpoints(url) onde estão hospedadas as páginas e o número de requisições HTTP por segundo pelo usuário. Para capturar esses dados em específico foi necessário utilizar o template “**Apache by Zabbix agent**”, que permite captar o tráfego específico de conteúdo web baseado em Apache.

Figura 132 - Gerenciamento Apache



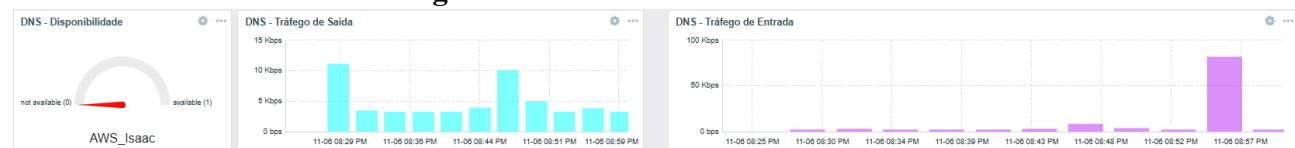
Fonte: Elaborado Pelo Grupo

Conforme foi possível verificar, o gráfico de disponibilidade indicava que a máquina do servidor web se encontrava indisponível no momento, com o marcador em 0, mas ainda sim foi possível analisar dados recebidos pelo Zabbix durante o período de atividade. Os gráficos apontam que a latência esteve mais alta durante o período em que as requisições por segundo aumentaram repentinamente, chegando a quase 200, e depois se estabilizou. Para fazer um teste de stress foi utilizado um comando para gerar várias requisições HTTP de uma única vez e analisar como o servidor se comportaria. A expectativa era de que ocorresse um congelamento do servidor, entretanto o Apache foi capaz de controlar a situação sem nem mesmo ativar o multiprocessamento, demonstrando sua capacidade.

3.1.2.4 Monitoramento do Servidor DNS

Os principais indicadores analisados para monitorar o tráfego do servidor DNS foram o tráfego de rede de entrada e saída de dados. Foram utilizados comandos de lookup para o domínio e verificar alterações nos gráficos.

Figura 133 - Gerenciamento do DNS



Fonte: Elaborado Pelo Grupo

Foi possível verificar que no momento a instância EC2 se encontra desligada,

mas os dados puderam ser verificados pelos registros nos momentos em que esteve ligada. Os gráficos registraram os bits enviados(saída) e recebidos(entrada) na VPC, que é uma interface de rede virtual privada criada pela AWS, onde o tráfego de rede da instância EC2 se encontra. Percebe-se que no geral o tráfego de saída é muito mais ativo do que o de entrada, um cenário um pouco incomum. Como a instância está utilizando um DNS recursivo.

3.1.2.5 Monitoramento do Servidor FTP

Os principais indicadores utilizados para analisar os dados do servidor FTP foram o tráfego de saída e entrada de dados na interface de rede, pois assim é possível analisar o comportamento da rede virtual privada da nuvem ou VPC durante a transferência de arquivos:

Figura 134 - Gerenciamento do FTP



Fonte: Elaborado Pelo Grupo

Como foi possível verificar, o servidor FTP enviou dados de saída mas pareceu receber dados de entrada em quantidades desprezíveis. Inicialmente foi pensado que poderia ser um problema do modo ativo de conexão, já que nele o canal de dados não é totalmente inicializado pelo cliente e possivelmente poderia causar interferência nos dados recebidos pelo servidor.

Para verificar a situação, primeiramente foi feito um teste para verificar se havia alguma identificação de bits recebidos por segundo(bps).

Figura 135 - Item de Bits recebidos

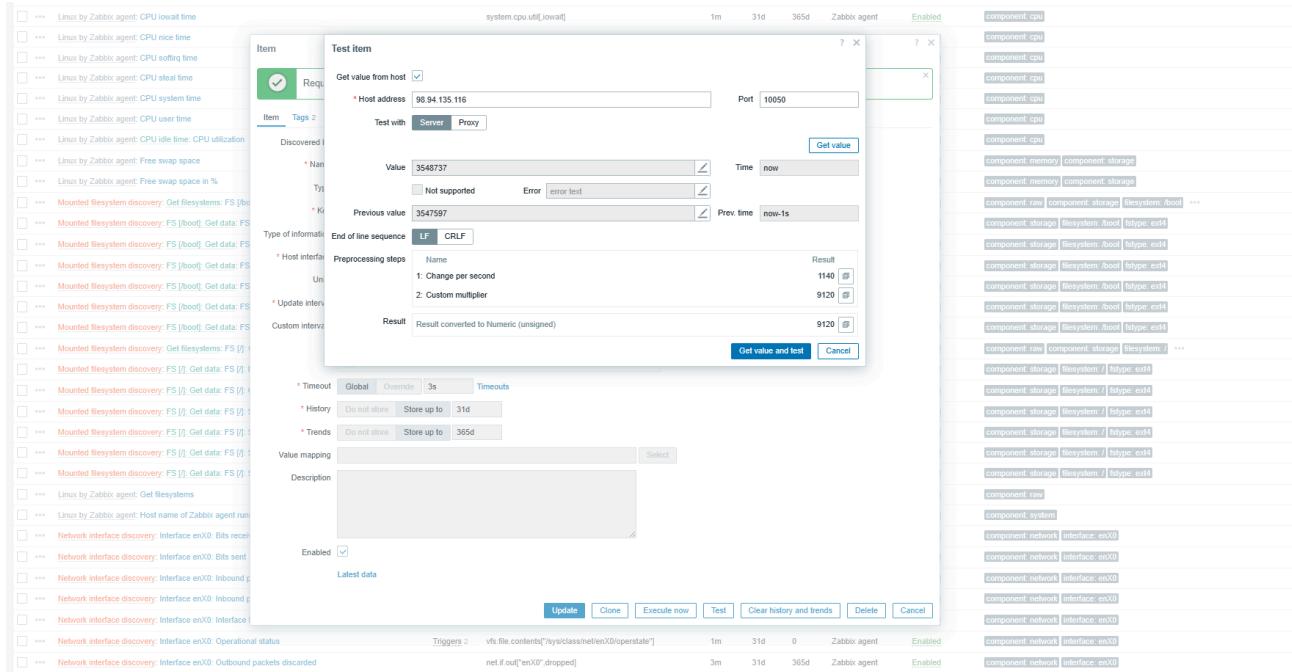
The screenshot shows the 'Item' configuration screen in Zabbix. The top status bar indicates 'Request sent successfully'. Below the title, there are tabs for 'Item', 'Tags 2', and 'Preprocessing 2'. The main configuration area includes:

- Discovered by:** Network interface discovery
- Name:** Interface enX0: Bits received
- Type:** Zabbix agent
- Key:** net.if.in["enX0"]
- Type of information:** Numeric (unsigned)
- Host interface:** 98.94.135.116:10050
- Units:** bps
- Update interval:** 3m
- Custom intervals:** A table showing one entry: Type: Flexible, Interval: 50s, Period: 1-7,00:00-24:00. An 'Add' button is available.
- Timeout:** Global (3s), Override (3s), Timeouts tab selected.
- History:** Do not store, Store up to 31d
- Trends:** Do not store, Store up to 365d
- Value mapping:** A field with a 'Select' button.
- Description:** A large text area.
- Enabled:** Checked (indicated by a checked checkbox).

At the bottom are buttons for **Update**, **Clone**, **Execute now**, **Test**, **Clear history and trends**, **Delete**, and **Cancel**.

Fonte: Elaborado Pelo Grupo

Figura 136 - Teste de Bits recebidos por segundo



Fonte: Elaborado Pelo Grupo

Havia um valor de bits sendo recebido e os pré-processadores do item apontam para mudanças ocorrendo. Com isso, há uma inconsistência entre o que o teste e o gráfico de entrada de bits apontam. Além disso, em “**Latest Data**” há a constante atualização de dados para o parâmetro de bits recebidos

Figura 137 - Últimos dados de bits recebidos



Fonte: Elaborado Pelo Grupo

Para tentar identificar problemas de recebimento de pacotes, foi habilitado o modo de conexão passiva nas configurações do vsftpd e foi utilizado o comando “**sudo tcpdump -i any 'port 21 or portrange 12000-12100' -n -A**” para identificar os pacotes recebidos ao estabelecer uma conexão com um cliente. O motivo da utilização de “**12000-12100**” foi porque esse foi o intervalo escolhido para estabelecer conexões no modo passivo.

Figura 138 - Pacotes do tráfego FTP nas portas 21 e dinâmicas do modo passivo

```

20:51:56.434918 enx0 Out IP 172.31.48.141.21 > 177.63.203.96.65256: Flags [P..], seq 165:215, ack 69, win 491, length 50: FTP: 227 Entering Passive Mode (98.94.135.116,47,26).
E..Z..@.r.@.^T..0...s3m...g.P...Y...227 Entering Passive Mode (98.94.135.116,47,26).

20:51:56.600582 enx0 In IP 177.63.203.96.65256 > 172.31.48.141.21: Flags [P..], seq 69:75, ack 215, win 1024, length 6: FTP: LIST
E...@.r.PV.?...0...g.s3m.P...DH..LIST

20:51:56.600807 enx0 In IP 177.63.203.96.65292 > 172.31.48.141.12058: Flags [S], seq 3596063065, win 65535, options [mss 1452,nop,wscale 7,nop,nop,sackOK], length 0
E..4..@.r.PV.?...0.../.W.Y......
20:51:56.600847 enx0 Out IP 172.31.48.141.12058 > 177.63.203.96.65292: Flags [S.], seq 4188483424, ack 3596063066, win 62727, options [mss 8961,nop,nop,sackOK,nop,wscale 7], length 0
E..4..@.r.W..0..?.../.W.Z...Ys...#.....
20:51:56.641062 enx0 Out IP 172.31.48.141.21 > 177.63.203.96.65256: Flags [.], ack 75, win 491, length 0
E..(.@.r.^...0...s3m...g.P...Yg..
20:51:56.748081 enx0 In IP 177.63.203.96.65292 > 172.31.48.141.12058: Flags [.], ack 1, win 32768, length 0
E..(.
@.r.PY.?...0.../.W.Z.../AP......
20:51:56.748235 enx0 Out IP 172.31.48.141.21 > 177.63.203.96.65256: Flags [P..], seq 215:254, ack 75, win 491, length 39: FTP: 150 Here comes the directory listing.
E..0..@.r.^...0...?...s3m...g.P...Y...150 Here comes the directory listing.

20:51:56.748306 enx0 Out IP 172.31.48.141.2058 > 177.63.203.96.65292: Flags [F.], seq 1, ack 1, win 491, length 0
E..()..@.r...0...?.../.A.W.ZP...Yg..
20:51:56.892725 enx0 In IP 177.63.203.96.65292 > 172.31.48.141.12058: Flags [.], ack 2, win 32768, length 0
E..(.@.r.PV.?...0.../.W.Z.../BP......
20:51:56.892830 enx0 Out IP 172.31.48.141.21 > 177.63.203.96.65256: Flags [P..], seq 254:278, ack 75, win 491, length 24: FTP: 226 Directory send OK.
E..0..@.r.^...0...s3m...g.P...OK...226 Directory send OK.

20:51:56.892949 enx0 In IP 177.63.203.96.65292 > 172.31.48.141.12058: Flags [F.], seq 1, ack 2, win 32768, length 0
E..(.@.r.PU.?...0.../.bP......
20:51:56.892970 enx0 Out IP 172.31.48.141.12058 > 177.63.203.96.65292: Flags [.], ack 2, win 491, length 0
E..(..@.r.^...0...?.../.b.W.[P......
20:51:56.952705 enx0 In IP 177.63.203.96.65256 > 172.31.48.141.21: Flags [.], ack 254, win 1023, length 0
E..(.@.r.PT.?...0...g.s3m.P......
20:51:57.082344 enx0 In IP 177.63.203.96.65256 > 172.31.48.141.21: Flags [.], ack 278, win 1023, length 0
E..(.@.r.PR.?...0...g.s3m.P......
20:52:04.451444 enx0 In IP 177.63.203.96.65256 > 172.31.48.141.21: Flags [P..], seq 75:81, ack 278, win 1023, length 6: FTP: PASV
E..@.r.PH.?...0...g.s3m.P...^...PASV

20:52:04.451496 enx0 Out IP 172.31.48.141.21 > 177.63.203.96.65256: Flags [.], ack 81, win 491, length 0
E..(..@.r.^...0...s3m...g.P...Yg..
20:52:04.451651 enx0 Out IP 172.31.48.141.21 > 177.63.203.96.65256: Flags [P..], seq 278:329, ack 81, win 491, length 51: FTP: 227 Entering Passive Mode (98.94.135.116,46,231).
E..[.0..@.r.^N..0..?...s3m...g.P...Y...227 Entering Passive Mode (98.94.135.116,46,231).

20:52:04.598569 enx0 In IP 177.63.203.96.65256 > 172.31.48.141.21: Flags [P..], seq 81:95, ack 329, win 1023, length 14: FTP: STOR HAHAX.TX
E..6..@.r.P?...0...g.s3n.P...g...STOR HAHAX.TX

20:52:04.598849 enx0 In IP 177.63.203.96.65294 > 172.31.48.141.12007: Flags [S], seq 157786691, win 65535, options [mss 1452,nop,wscale 7,nop,nop,sackOK], length 0
E..4..@.r.P@.?...0...g.C...<......
20:52:04.598906 enx0 Out IP 172.31.48.141.12007 > 177.63.203.96.65294: Flags [S.], seq 3327927871, ack 157786692, win 62727, options [mss 8961,nop,nop,sackOK,nop,wscale 7], length 0
E..4..@.r.W..0..?...\\& g.0...Ys...#.....

```

Fonte: Elaborado Pelo Grupo

Como é possível ver, o servidor FTP está recebendo e enviando pacotes através da conexão feita com um cliente. A interface de rede demonstra através de “Out” e “In” que tanto o envio quanto o recebimento de pacotes pelo servidor ocorrem. Portanto, não foi possível identificar um motivo plausível para que o frontend Zabbix não indique esses dados no dashboard.

3.2 Evidências do Monitoramento dos Serviços VirtualBox

3.2.1 Ambiente On-Premise (VirtualBox)

Para o ambiente on-premise, que hospeda o serviço de DHCP, foi implementada uma instância dedicada do Zabbix para monitorar tanto o servidor Ubuntu quanto o cliente Windows.

3.2.2. Implementação do Servidor Zabbix

O processo iniciou-se com a importação do Zabbix Appliance para o VirtualBox. Para permitir tanto o acesso à sua interface web quanto a comunicação com a rede interna dos serviços, o servidor Zabbix foi configurado com duas placas de rede: a primeira em modo Bridge, para obter um endereço IP da rede local (192.168.1.250), e a segunda em modo Rede Interna (intnet), com um endereço IP estático (192.168.99.10) para se comunicar com os hosts a serem monitorados.

3.2.3 Configuração dos Agentes de Monitoramento

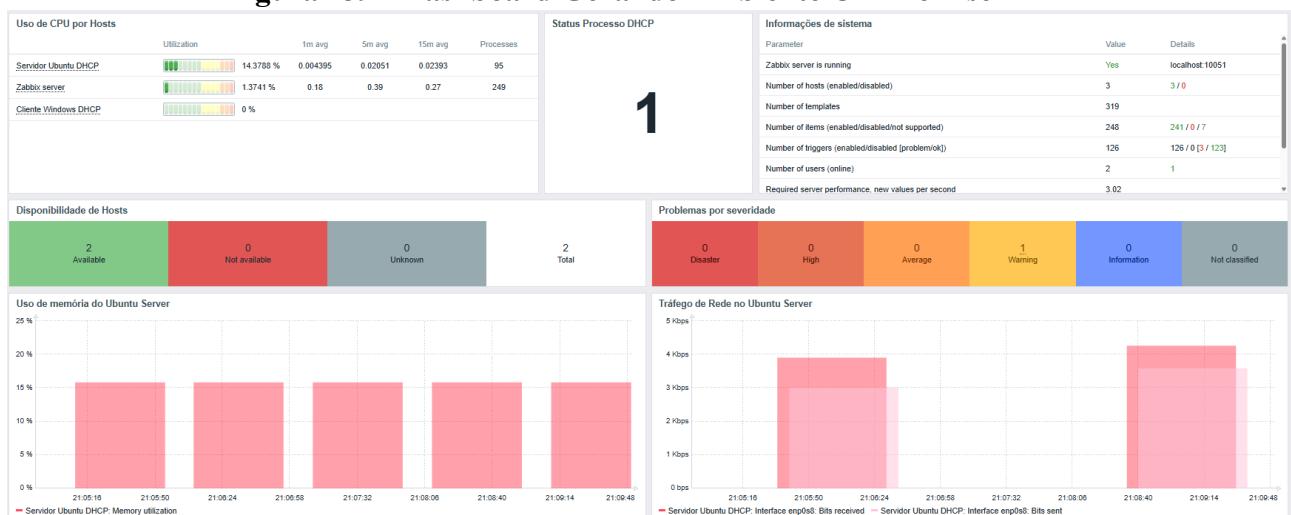
Para que o Zabbix pudesse coletar dados, foi necessário preparar cada host-alvo:

- Servidor Ubuntu DHCP:** Foi instalado o zabbix-agent e seu arquivo de configuração (zabbix_agentd.conf) foi ajustado para permitir conexões vindas do IP interno do servidor Zabbix (192.168.99.10).
- Cliente Windows Server:** Foi habilitado o Protocolo SNMP (Simple Network Management Protocol) nativo do Windows. Na configuração do serviço, foi criada a comunidade “public” com permissão de READ ONLY e foi configurado para aceitar requisições SNMP exclusivamente do IP interno do servidor Zabbix (192.168.99.10), garantindo a segurança do monitoramento.

3.2.4. Resultados do Monitoramento

Após a configuração dos hosts na interface web do Zabbix, foi criado um dashboard personalizado para centralizar as informações mais relevantes do ambiente on-premise.

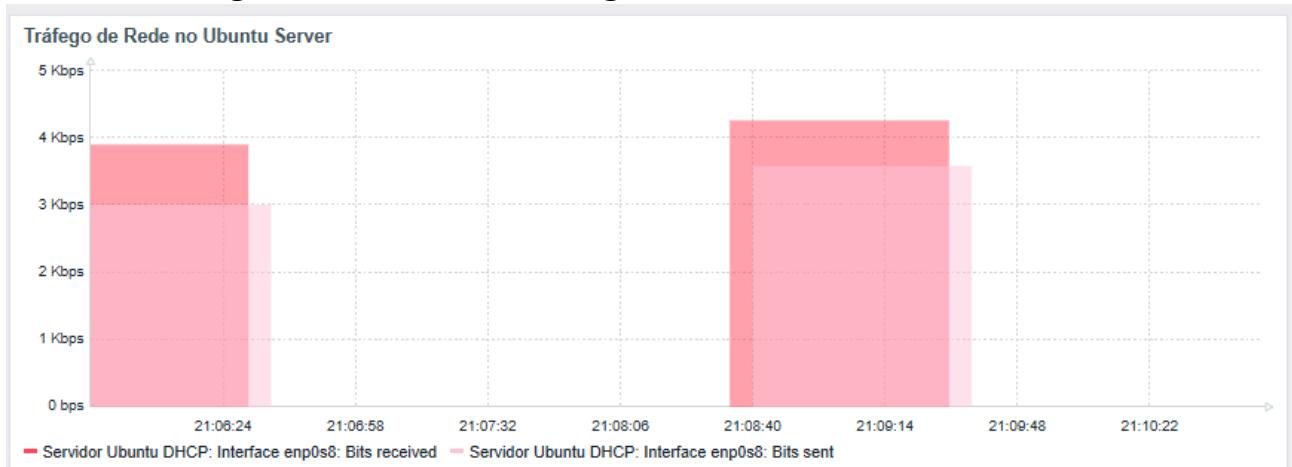
Figura 139 - Dashboard Geral do Ambiente On-Premise



Fonte: Elaborado Pelo Grupo

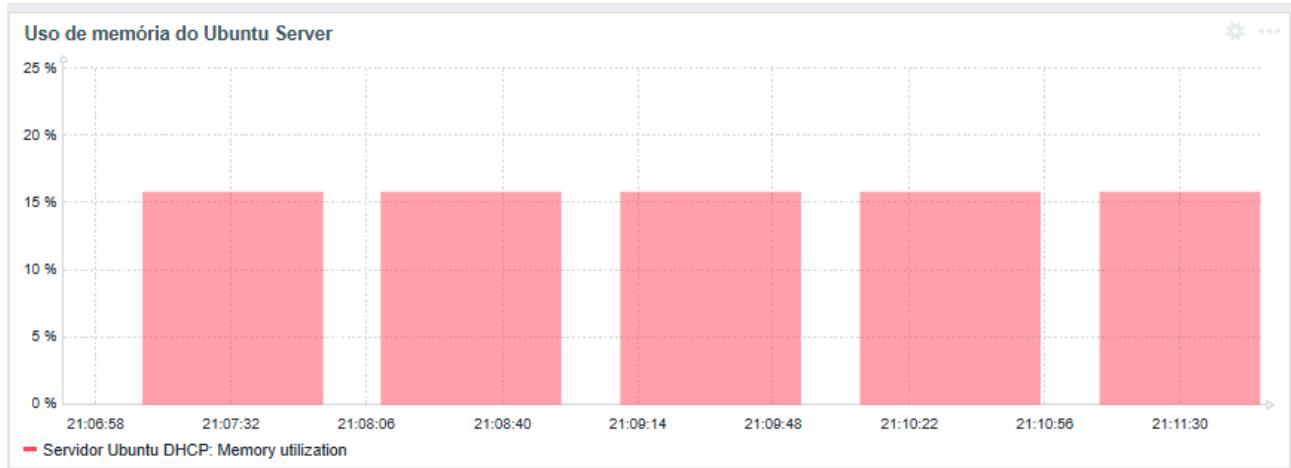
O painel centraliza informações vitais como a disponibilidade dos hosts, o uso de CPU, e um widget específico para o status do processo DHCP, que confirma que o serviço está ativo e em execução. Além do dashboard, foram gerados gráficos específicos para analisar o comportamento detalhado dos serviços, como o tráfego de rede, que evidencia a comunicação entre cliente e servidor DHCP, e o uso de recursos de hardware ao longo do tempo.

Figura 140 - Gráfico de Tráfego de Rede no Servidor Ubuntu



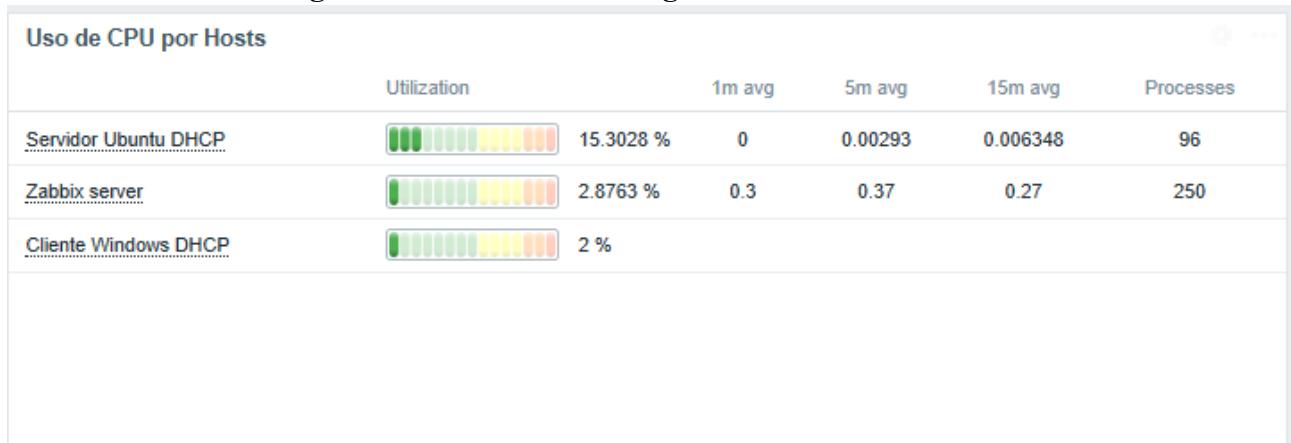
Fonte: Elaborado Pelo Grupo

Figura 141 - Gráfico de Uso de Memória no Servidor Ubuntu



Fonte: Elaborado Pelo Grupo

Figura 142 - Gráfico de Carga de CPU dos hosts



Fonte: Elaborado pelo grupo.

A implementação bem-sucedida do Zabbix forneceu uma visão clara e em tempo real da saúde e do desempenho do ambiente on-premise, cumprindo os objetivos propostos para esta etapa do projeto.

ETAPA 4 – MECANISMOS DE SEGURANÇA DA INFORMAÇÃO

Nesta etapa, são abordadas medidas essenciais de proteção para os recursos computacionais da Solo Forte Agropecuária, visando assegurar a confidencialidade, integridade e disponibilidade das informações. São apresentadas a Política de Segurança da Informação (PSI), uma cartilha com orientações aos colaboradores, e a análise do sistema CRUD desenvolvido em ASP.NET, com base nas vulnerabilidades mais recorrentes da OWASP Top 10.

4.1 Política de Segurança da Informação (PSI)

A Solo Forte Agropecuária adota uma Política de Segurança da Informação estruturada, que estabelece diretrizes para proteção dos dados corporativos, assegurando a confidencialidade, integridade e disponibilidade das informações utilizadas em suas operações. Essa política define os processos relacionados ao gerenciamento de acessos físicos e lógicos, segurança de redes, monitoramento, continuidade de negócios e conformidade com normas e legislações aplicáveis.

Entre os controles previstos, destacam-se o gerenciamento de credenciais de acesso, uso de mecanismos de autenticação, registro e monitoramento de atividades em sistemas, proteção física de ambientes críticos, aplicação de planos de contingência para falhas operacionais e abordagem contínua de treinamento e conscientização dos colaboradores sobre boas práticas de segurança. Além disso, são estabelecidas responsabilidades formais para a alta direção, equipe técnica e demais funcionários, garantindo que todos estejam alinhados às diretrizes de proteção dos ativos informacionais da organização.

A PSI, ao considerar também a conformidade com legislações como a LGPD, reforça o compromisso da Solo Forte para com a preservação da privacidade e gestão responsável dos dados de clientes, parceiros e funcionários. Dessa forma, a estrutura de Segurança da Informação contribui diretamente para a continuidade operacional e para a confiança institucional no ambiente do agronegócio.

4.2 Cartilha de Boas Práticas de Segurança da Informação

Para reforçar a aplicação da PSI, foi criada uma cartilha com medidas práticas de segurança a serem adotadas pelos funcionários no dia a dia, incluindo:

- Não compartilhar credenciais de acesso.
- Evitar abrir anexos e links suspeitos em e-mails.
- Bloquear a estação de trabalho ao se ausentar do posto.
- Utilizar exclusivamente dispositivos autorizados em rede interna.
- Manter o sigilo sobre informações sensíveis da empresa.
- Reportar incidentes ou atividades suspeitas imediatamente ao setor de TI.
- Não utilizar redes Wi-Fi públicas para acessar sistemas corporativos.

Essas orientações reforçam a cultura de segurança e reduzem o risco de ameaças internas e externas.

4.3 Análise de Vulnerabilidades do CRUD com base na OWASP

A aplicação CRUD desenvolvida em ASP.NET Core foi analisada considerando os riscos apresentados pela OWASP Top 10, que lista as vulnerabilidades mais críticas em aplicações web. Com base nisso, foram identificados riscos potenciais e definidos controles de mitigação adequados ao contexto do sistema.

Tabela de conformidade às 10 principais vulnerabilidades da OWASP

Vulnerabilidade	Risco	Evidências
A01:2021 - Quebra de Controle de Acesso	Crítico	Não há um fluxo de login ou autenticação do usuário. Não há exigência de autorização para manipular os dados da API.
A02:2021 - Falhas Criptográficas	Baixo	O ambiente em produção é criptografado com certificado SSL para ambos o backend e o frontend. Interface não inclui nenhum dado sensível do cliente, visto que não há sistema de login e/ou cadastro.
A03:2021 - Injeção	Alto	Não há limites de caracteres para os campos do produto. A interface permite ao usuário criar um produto sem nenhuma unidade disponível. Não há validação para criação de produtos com mesmo nome. Uso padronizado de medidas que automaticamente parametrizam consultas ao banco como medida contra injeção de código SQL malicioso.
A04:2021 - Design Inseguro	Baixo	Arquitetura do backend segura e padronizada, segundo princípios de acoplamento e responsabilidade estabelecidos pelo padrão SOLID. As operações de criar e excluir retornam mensagens sobre o status do produto-alvo.
A05:2021 - Configuração Insegura	Baixo	Simplicidade funcional que limita as opções acessíveis ao usuário final.
A06:2021 - Componente Desatualizado e Vulnerável	Médio	Versão de ferramentas do backend antiga, constando como .NET 8.0. Interface do usuário construída em cima de ferramenta atualizada e moderna, com compilador Vite e biblioteca React na penúltima versão 18.

Tabela de conformidade às 10 principais vulnerabilidades da OWASP

A07:2021 - Falha de Identificação e Autenticação	Alto	O sistema não possui suporte para autenticar usuários, portanto o acesso se faz fatidicamente anônimo. Acesso livre às informações da API, entretanto o banco de dados responsável por gerenciá-las é restrito a nível institucional.
A08:2021 - Falha na Integridade de Dados e Software	Crítico	Indisponibilidade de serviços com fins de mantinabilidade. Falta de fluxos para verificação de vulnerabilidades por versionamento. Ausência de padrões para identificação de deploys.
A09:2021 - Monitoramento de Falhas e Registros de Segurança	Alto	Ausência de alertas para detecção de atividades suspeitas. Falta de automatização de processos para identificação de erros sobre insegurança.
A10:2021 - Falsificação de Solicitação do Lado do Servidor	Baixo	A API não possui chamadas internas configuradas de acordo com cargos. Domínio e Cross-Origin Resource-Sharing devidamente configurados para aceitar somente requisições internas do cliente, não possibilitando chamadas externas.

ETAPA 5 – APRESENTAÇÃO FINAL E RELATÓRIO TÉCNICO DO PROJETO DE REDES

Durante a etapa 5, foi desenvolvida uma apresentação no PowerPoint abordando os aspectos gerais do percurso de desenvolvimento do projeto Solo Forte Agropecuária e a proposta geral da implementação de uma rede para compor tal. Ao final, foram levantados tópicos tratando-se das principais dificuldades enfrentadas durante esse período.

CONCLUSÃO

Este estudo teve como objetivo simular e demonstrar o processo de implementação da rede em acordo com os requisitos de uma empresa fictícia de agropecuária. Através das medidas tomadas, constatou-se que o desenvolvimento satisfaz as condições básicas necessárias. Portanto os resultados obtidos refletem-se para um ambiente ficcional, contudo para casos de uso reais o projeto teria que ser futuramente incrementado em sua robustez e capacidade tecnológica.

Alguns problemas de arquitetura da rede dimensionada não foram solucionados ou refatorados para o alinhamento às expectativas desejadas, embora a maioria do trabalho tenha representado situações de êxito.

A pesquisa permitiu identificar tópicos importantes a serem abordados na hora de se pensar na construção de uma rede corporativa e como é a inicialização para se constituir uma infraestrutura telecomunicacional e tecnológica de infraestrutura de rede em uma empresa.

REFERÊNCIAS

ANICAS, M.; ELLINGWOOD, J. **Como configurar o BIND como um servidor DNS de rede privada no Ubuntu 18.04.** DigitalOcean, 2020. Disponível em:<<https://www.digitalocean.com/community/tutorials/how-to-configure-bind-as-a-private-network-dns-server-on-ubuntu-18-04-pt>>. Acesso em: 28 nov. 2025.

BAKALDIN, A.; BIBA, A.; GORDYMOV, EV et al. **Apache by Zabbix Agent.** Zabbix, 2025. Disponível em:<https://github.com/zabbix/zabbix/tree/release/7.4/templates/app/apache_agent#apache-by-zabbix-agent>. Acesso em: 28 nov. 2025.

BAKALDIN, A.; RASIKHOV, D.; KLOPOVSKY, Y. **PostgreSQL by Zabbix Agent.** Zabbix, 2025. Disponível em:<https://github.com/zabbix/zabbix/tree/release/7.4/templates/db/postgresql_agent>. Acesso em: 28 nov. 2025.

CISCO NETWORKING ACADEMY. **Começando com o Cisco Packet Tracer.** Disponível em:<<https://www.netacad.com/courses/getting-started-cisco-packet-tracer?courseLang=pt-BR>>.

Acesso em: 27 nov. 2025.

CORDEIRO, F. L. R. **Aula 1: Configuração Roteamento RIP2 com Packet Tracer.** Youtube, 2023. Disponível em: <<https://www.youtube.com/watch?v=VCorikUaoIQ&t=67s>>. Acesso em: 27 nov. 2025.

CORDEIRO, F.L.R. **Aula 2: Configuração Roteamento RIP2 com Packet Tracer.** Youtube, 2023. Disponível em: <<https://www.youtube.com/watch?v=WVr6L8LeUD4>>. Acesso em: 27 nov. 2025.

CORDEIRO, F.L.R. **Aula 3: Configuração Roteamento RIP2 com Packet Tracer.** Youtube, 2023. Disponível em: <<https://www.youtube.com/watch?v=9m9swyaZwfo>>. Acesso em: 27 nov. 2025.

CORDEIRO, F.L.R. **Cloud AWS Learner Lab : Aula 1 - Introdução IaaS.** Youtube, 2023. Disponível em: <<https://www.youtube.com/watch?v=tercmQtTi0c&t=11s>>. Acesso em: 27 nov. 2025.

CORDEIRO, F.L.R. **Cloud AWS Learner Lab: Aula 2 - Criando uma VM e fazendo conexão SSH.** Youtube, 2023. Disponível em: <<https://www.youtube.com/watch?v=PB5HLpv-GjM&t=29s>>. Acesso em: 28 nov. 2025.

CORDEIRO, F.L.R. **Cloud AWS Learner Lab : Aula 3 - Instalando Servidor Apache e Hospedando Site.** Youtube, 2023. Disponível em: <<https://www.youtube.com/watch?v=MTLNcIuIEns&t=47s>>. Acesso em: 28 nov. 2025.

CORDEIRO, F.L.R. **Configuração de AD e GPO.** Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte, 2025.

CORDEIRO, F.L.R. **Projeto: Infraestrutura de Redes:** Pratica 02: Instalação e configuração de FTP e Servidor Web no Ubuntu AWS EC2. Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte, 2025.

CORDEIRO, F.L.R. **Eixo 5: Projeto de Infraestrutura de Rede:** Prática 3 - Instalação e configuração do Linux no Virtual Box. Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte, 2025.

CORDEIRO, F.L.R. **Modelo de artigo do Instituto de Ciências Exatas e de Informática.** Modelo de Artigo - ICEI PUC Minas (vs2024). Overleaf, Brasil, 2024. Disponível em: <<https://www.overleaf.com/latex/templates/modelo-de-artigo-icei-puc-minas-vs-2024/rpkkgvrybtvc>>. Acesso em: 27 nov. 2025.

CORDEIRO, F.L.R. **[Identificação de Software para Gerenciamento de Computadores].** Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte, p. 1-13, 2025.

CORDEIRO, F.L.R. **tutorial AWS.** Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte, p. 7-18, 2025.

CORDEIRO, F.L.R. **Segurança de Sistemas de Informação:** Aula 6: Politicas de Segurança da Informação. Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte, p. 1-21, 2025.

DOSANJH, S. **Understanding PostgreSQL's Cache Hit Ratio:** This article explores PostgreSQL's buffer cache hit ratio (BCHR), a metric tracked by Redgate Monitor to help you assess the health and performance of your databases. By correctly interpreting this ratio and using it with other relevant memory and IO metrics, you can identify potential bottlenecks in query execution before they impact users. Redgate Monitor, 2024. Disponível em:<<https://www.red-gate.com/hub/product-learning/redgate-monitor/understanding-postgresql-s-cache-hit-ratio>>. Acesso em: 28 nov. 2025.

LOVETT, C. **Understanding CapEx vs OpEx Cloud Services.** CapEx vs. OpEx Cloud: What's The Difference? Tierpoint, 2023. Disponível em <<https://www.tierpoint.com/blog/capex-vs-opex-cloud-whats-the-difference/>>. Acesso em: 28. nov 2025.

ORACLE, VirtualBox. **Download VirtualBox:** The VirtualBox Extension Pack is available for personal and educational use on this page under the PUEL license. The VirtualBox Extension Pack is also available under commercial or enterprise terms. By downloading, you agree to the terms and conditions of the respective license. Disponível em <<https://www.virtualbox.org/wiki/Downloads>>. Acesso em: 28 nov. 2025.

OWASP. **What's changed in the Top 10 for 2021.** Welcome to the OWASP Top 10 - 2021, OWASP, p. 2, 2021. Disponível em: <https://owasp.org/Top10/A00_2021_Introduction/> Acesso em: 28. nov 2025.

RASHID, MUHAMMAD. **Deploy Frontend and Backend on Same Server - AWS EC2 instance - Django and React.** Youtube, 2024. Disponível em: <<https://www.youtube.com/watch?v=UVnxBDasJT4>>. Acesso em: 27. nov 2025.

WILLIAMSON, JANILLA. **How to host a website using Apache in an AWS Instance.** AWS Plain English, 2022. Disponível em: <<https://aws.plainenglish.io/how-to-host-a-website-using-apache-in-an-aws-instance-for-some-what-of-a-beginner-e10d80c0ff7d>>. Acesso em: 27 nov. 2025.

ZABBIX. **7 Configuração.** [S. l.]: 7.4 ed. Documentação Zabbix, p. 7-8, 2025. Disponível em: <<https://www.zabbix.com/documentation/current/pt/manual/config>>. Acesso em: 28 nov. 2025.