



PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Instituto de Ciências Exatas e de Informática

Modelo de artigo do Instituto de Ciências Exatas e de Informática*

Alice Abreu dos Reis¹
Gabriel dos Reis Nascimento²
Leandro Augusto Santos Araujo³
Martha Beatriz Siqueira da Silva⁴
Omar Petronilio Martins de Abreu⁵
Fabio Leandro Rodrigues⁶

Resumo

Este projeto detalha a LOGAM Tech, uma empresa de telemarketing fundada em 2021 , especializada no nicho de tecnologia. Sua missão é conectar indivíduos à educação em tecnologia, oferecendo atendimento humano e eficiente , suporte em vendas (B2B, B2C e B2G), cobrança e orientação personalizada para cursos de parceiros como Alura e Rocket-seat. A empresa opera a partir do Rio de Janeiro (sede) e possui filiais operacionais em Brasília, Curitiba, Salvador e Belém , com atuação em todo o território nacional. O escopo técnico do projeto define a arquitetura de rede para suportar as operações multicanal da LOGAM Tech , que dependem de sistemas robustos de VOIP, CRM, ERP e uma infraestrutura segura para o trabalho remoto/híbrido via VPN. Foi adotada uma topologia em estrela para facilitar o gerenciamento e garantir o isolamento de falhas entre a sede e as filiais. A sede no Rio de Janeiro (faixa 192.168.0.0/24) centraliza serviços essenciais como DHCP, FTP, HTTP e DNS e utiliza VLANs para segregação de tráfego entre Funcionários e Visitantes. A visão da empresa é ser a maior referência nacional em orientação para cursos de tecnologia até 2030, expandindo globalmente.

Palavras-chave: Telemarketing; Tecnologia; Rede de Computadores.

*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.

¹Aluno(a) do Programa de Graduação em Sistemas de Informação – alice.reis.954530@sga.pucminas.br.

²Aluno(a) do Programa de Graduação em Sistemas de Informação – xxx@sga.pucminas.br.

³Aluno(a) do Programa de Graduação em Sistemas de Informação – xxx@sga.pucminas.br.

⁴Aluno(a) do Programa de Graduação em Sistemas de Informação – xxx@sga.pucminas.br.

⁵Aluno(a) do Programa de Graduação em Sistemas de Informação – omar.abreu@sga.pucminas.br.

⁶Professor(a) do Programa de Graduação em Sistemas de Informação – xxx@pucminas.br.

Abstract

This project details LOGAM Tech, a telemarketing company founded in 2021, specializing in the technology niche. Its mission is to connect individuals to technology education, offering efficient and human service, sales support (B2B, B2C, and B2G), collection, and personalized guidance for courses from partners such as Alura and Rocketseat. The company operates from its headquarters in Rio de Janeiro and has operational branches in Brasília, Curitiba, Salvador, and Belém, serving the entire national territory. The technical scope of the project defines the network architecture needed to support LOGAM Tech's multi-channel operations, which rely on robust VOIP, CRM, and ERP systems. A secure infrastructure is also required for remote/hybrid work via VPN. A star topology was adopted to facilitate management and ensure fault isolation between the headquarters and the branches. The Rio de Janeiro headquarters (network range 192.168.0.0/24) centralizes essential services such as DHCP, FTP, HTTP, and DNS, and utilizes VLANs to segregate traffic between Employees and Visitors. The company's vision is to become the largest national reference in technology course guidance by 2030, pursuing global expansion.

Keywords: Telemarketing; Technology; Computer Network.

1 INTRODUÇÃO

A LOGAM Tech opera a partir de sua sede no Rio de Janeiro, RJ, com filiais em Brasília, Curitiba, Salvador e Belém. Sua atuação se estende a todo o território nacional por meio de modelos de trabalho remoto e híbrido, o que possibilita maior alcance e flexibilidade operacional. O presente trabalho tem como objetivo analisar a infraestrutura tecnológica da empresa e propor soluções de rede que sustentem suas operações de telemarketing especializado em tecnologia, garantindo desempenho, segurança e disponibilidade dos serviços.

1.1 Ramo de atividade

A LOGAM Tech atua no setor de Telemarketing e Contact Center, com especialização no nicho de tecnologia. Seu foco está na divulgação, vendas (B2B, B2C e B2G), suporte técnico, cobrança e orientação para cursos de capacitação na área de tecnologia. A empresa busca oferecer um atendimento consultivo, aproximando pessoas das oportunidades de formação que melhor atendam seus objetivos profissionais.

1.2 Micro-história

Fundada em 2021, a LOGAM Tech surgiu em um contexto de acelerada demanda por profissionais qualificados na área de tecnologia. A empresa foi criada a partir da percepção de que havia uma lacuna entre pessoas interessadas em requalificação profissional e as instituições que ofereciam cursos na área de TI. Dessa forma, sua proposta não é apenas comercializar cursos de diferentes escolas, mas orientar os estudantes na escolha do caminho mais adequado para seu desenvolvimento profissional. Com uma abordagem humana e tecnológica, a LOGAM Tech valoriza seus atendentes por meio de treinamentos contínuos e políticas de trabalho flexíveis, incluindo o home office. A empresa também investe em ferramentas modernas de comunicação e gestão de clientes, consolidando-se como parceira de grandes escolas de tecnologia, como a Alura e a Rocketseat, impactando positivamente milhares de pessoas em todo o país.

1.3 Missão

A missão da LOGAM Tech é conectar pessoas à educação em tecnologia por meio de um atendimento eficiente, humanizado e orientado a resultados, oferecendo suporte para que cada indivíduo encontre o curso ideal para seu crescimento profissional.

1.4 Visão

Ser, até 2030, a maior referência nacional em atendimento e orientação para cursos de tecnologia, unindo inovação tecnológica à excelência no relacionamento humano, e expandindo gradualmente sua atuação para o cenário internacional.

1.5 Estrutura de serviços

A atuação da LOGAM Tech está organizada em quatro frentes estratégicas, que visam atender de forma completa seus clientes e parceiros:

Vendas e prospecção: realização de vendas ativas e receptivas (B2B, B2C e B2G), prospecção de clientes e orientação personalizada na escolha de cursos e planos de carreira.

Cobrança: equipes especializadas na gestão de pagamentos e negociação de débitos, com foco na retenção de alunos e na recuperação de créditos.

Suporte técnico: assistência personalizada a alunos e empresas parceiras em relação a plataformas e sistemas de ensino utilizados.

Orientação educacional: direcionamento de estudantes com base em seus perfis e objetivos, fortalecendo a experiência do cliente.

1.6 Compromisso social e sustentabilidade

Em alinhamento com o Objetivo de Desenvolvimento Sustentável (ODS) 4: Educação de Qualidade, a LOGAM Tech mantém parcerias com instituições de ensino que oferecem cursos de tecnologia gratuitos ou a baixo custo para pessoas de baixa renda. Essa iniciativa tem como propósito promover a inclusão digital e ampliar o acesso à capacitação profissional em comunidades vulneráveis, contribuindo para a formação de novos talentos no mercado de TI.

1.7 Impacto dos serviços na infraestrutura

Os serviços oferecidos pela LOGAM Tech exercem um impacto direto sobre a infraestrutura de rede corporativa, exigindo planejamento técnico adequado para garantir desempenho, segurança e disponibilidade. O modelo de atendimento multicanal depende de servidores VOIP e CRM robustos, capazes de gerenciar grandes volumes de chamadas e interações, além de requerer conectividade estável com a internet e mecanismos de proteção de dados sensíveis. As operações de vendas e prospecção demandam alta disponibilidade de rede para assegurar a fluidez da comunicação, enquanto o processamento e a análise de dados de clientes dependem de bancos de dados consistentes, com backups regulares e acesso remoto seguro. O setor de

cobrança, por sua vez, utiliza um ERP financeiro integrado ao CRM, garantindo controle de inadimplência e segurança das informações. Além disso, o modelo híbrido e remoto adotado pela empresa requer o uso de uma VPN corporativa, assegurando que os atendentes possam acessar os sistemas internos de forma protegida e confiável.

1.8 Identidade visual

A identidade visual da LOGAM Tech utiliza cores como preto, verde e cinza, que representam modernidade, inovação e confiança — valores associados ao setor tecnológico e ao posicionamento da marca.

Figura 1 – Logo - LOGAM Tech



LOGAM Tech
Conectando Pessoas à Tecnologia

Fonte: Elaborado pelos autores

2 PROTÓTIPO DO PROJETO DO CISCO PACKET TRACER

2.1 Topologia

A topologia em estrela foi escolhida por oferecer maior facilidade de gerenciamento e manutenção. Cada filial possui sua própria rede em estrela conectada à sede, o que garante isolamento de falhas locais, ou seja, se um switch de uma filial falhar, as demais continuam funcionando normalmente. A centralização no roteador da sede permite maior controle administrativo e segurança, simplificando a comunicação entre filiais e reduzindo a complexidade da rede.

2.2 Descrição da sede (Rio de Janeiro)

Na sede foram configurados diversos serviços essenciais para toda a rede corporativa. O servidor principal concentra os serviços de DHCP, FTP, HTTP e DNS, oferecendo maior controle e eficiência.

- A faixa de rede utilizada na sede é 192.168.0.0/24, tendo como gateway o endereço 192.168.0.1.
- O serviço DHCP distribui endereços automaticamente no intervalo de 192.168.0.10 a 192.168.0.99, facilitando a administração dos dispositivos conectados.
- O serviço DHCP distribui endereços automaticamente no intervalo de 192.168.0.10 a 192.168.0.99, facilitando a administração dos dispositivos conectados.

A rede foi estruturada em duas VLANs, separando Funcionários e Visitantes a fim de garantir maior segurança e controle. O roteador foi configurado para gerenciar ambas as VLANs por meio de um único link com o switch, enquanto um roteador wireless atende aos Funcionários e um Access Point atende aos Visitantes, ambos utilizando conexão WPA2. No servidor DHCP, foram configurados dois pools distintos, tanto para as filiais quanto para a sede, assegurando a distribuição correta dos IPs.

2.3 Descrição de uma filial (Curitiba)

Na filial de Curitiba, foi configurado um servidor dedicado apenas ao serviço DHCP, responsável por atribuir endereços IP dentro da faixa da unidade (10.4.0.0/24). Assim como na sede, a filial conta com Access Points configurados para oferecer conexão Wi-Fi a funcionários e visitantes, possibilitando mobilidade sem comprometer a segurança da rede. Os computadores da filial estão distribuídos entre os setores de Administração, Cobrança e Vendas, todos conectados ao switch central. Esse switch concentra o tráfego e encaminha ao roteador local, que faz a comunicação com a sede no Rio de Janeiro.

3 PREPARAÇÃO DO AMBIENTE EM NUVEM E VIRTUALIZAÇÃO LOCAL

Nesta etapa foram configurados os principais serviços de rede e infraestrutura da LOGAM Tech, distribuídos entre ambientes em nuvem (AWS) e locais (VirtualBox). As implementações realizadas nesta fase têm como objetivo garantir autenticação centralizada, comunicação segura, controle de endereçamento automático, hospedagem de dados e disponibilidade dos serviços corporativos.

3.1 Active Directory (AD), DNS e GPO

O servidor Active Directory (AD) foi configurado em uma instância hospedada na AWS, com o objetivo de centralizar a autenticação e o gerenciamento de usuários e dispositivos da LOGAM Tech. O DNS foi implementado de forma integrada ao AD, possibilitando a resolução de nomes e a comunicação eficiente entre os servidores e clientes da rede. Além disso, foram criadas políticas de grupo (GPO) para aplicar configurações de segurança e controle de acesso aos usuários, garantindo padronização no ambiente corporativo.

Tabela 1 – Informações do servidor (AWS)

Nome do Servidor	Endereço IP Público	Endereço IP Privado	Usuário de Acesso	Função
dc1-puc	52.23.39.125	10.0.1.162	N/A	Centraliza autenticação e gerenciamento de usuários via AD, com DNS e GPO integrados.

Fonte: Criado pelos autores

3.2 Configuração de Segurança (Security Groups)

3.2.1 *Regras de Entrada — group-sg-ad*

- Descrição: Define o tráfego permitido para o Controlador de Domínio (AD, DNS, Kerberos, LDAP, SMB).
- Objetivo: Permitir autenticação, replicação, resolução de nomes e aplicação de políticas entre máquinas da VPC 10.0.0.0/16.

Tabela 2 – Regras de entrada — group-sg-ad

Tipo	Protocolo	Porta / Intervalo	Origem	Descrição
UDP personalizado	UDP	464	10.0.0.0/16	Autenticação Kerberos (UDP)
TCP personalizado	TCP	88	10.0.0.0/16	Kerberos (TCP)/ autenticação segura
LDAP	TCP	389	10.0.0.0/16	Diretório e autenticação LDAP
SMB	TCP	445	10.0.0.0/16	Compartilhamento SYSVOL e NETLOGON
DNS (UDP)	UDP	53	0.0.0.0/0	Resolução DNS (UDP)
SSH	TCP	22	0.0.0.0/0	Acesso remoto
ICMP	ICMP	Tudo	10.0.0.0/16	Ping e diagnóstico interno

3.2.2 Regras de Saída — group-sg-ad

- Todos os protocolos: 0 . 0 . 0 . 0 / 0 — Comunicação livre.

3.2.3 Regras de Entrada — group-sg-client

Tabela 3 – Regras de entrada — group-sg-client

Tipo	Protocolo	Porta	Origem	Descrição
SSH	TCP	22	0.0.0.0/0	Acesso remoto (Linux)
RDP	TCP	3389	[IP Público do ADM]	Acesso remoto (Windows) via IP fixo

- Saída: Todos os protocolos (0 . 0 . 0 . 0 / 0) — Comunicação livre.

3.3 Configuração DNS — Route 53 (Zona Privada)

- Descrição: Zona hospedada privada corp.logamtech.local utilizada para resolução interna entre as instâncias da VPC.

Tabela 4 – Registros DNS

Nome	Tipo	Valor	Função
ftp.corp.logamtech.local	A	18.212.95.192	Servidor FTP
web-server.corp.logamtech.local	A	54.145.137.231	Servidor Web

3.4 Criar a Instância EC2 do Controlador de Domínio

- Nome: dc1-puc
- Sistema Operacional: Ubuntu Server 22.04 LTS
- Tipo de Instância: t3.micro
- Security Group: group-sg-ad

3.4.1 *Elastic IP*

- Objetivo: Garantir IP fixo para o DC.
- Elastic IP: 52.23.39.125
- Instância: dc1-puc

3.5 Configuração do Servidor AD/DC

3.5.1 *Acesso à instância*

```
1 ssh -i "[chave-ssh]" ubuntu@52.23.39.125
```

3.5.2 *Atualização dos pacotes*

```
1 sudo apt update && sudo apt upgrade -y
```

3.5.3 *Instalação de dependências*

```
1 sudo apt install samba krb5-config winbind smbclient dnsutils ldb-tools ntp  
-y
```

3.5.4 *Configuração do Kerberos*

```
1 Realm: CORP.LOGAMTECH.LOCAL
2 KDC: dcl.corp.logamtech.local
3 Admin Server: dcl.corp.logamtech.local
```

3.6 Provisionamento do Samba AD/DC

```
1 sudo mv /etc/samba/smb.conf /etc/samba/smb.conf.bak
2 sudo samba-tool domain provision \
3   --realm=CORP.LOGAMTECH.LOCAL \
4   --domain=CORP \
5   --server-role=dc \
6   --dns-backend=SAMBA_INTERNAL \
7   --use-rfc2307
8 sudo cp /var/lib/samba/private/smb.conf /etc/samba/smb.conf
```

3.6.1 Atualizar DNS local

```
1 sudo nano /etc/resolv.conf
2 nameserver 127.0.0.1
3 search corp.logamtech.local
```

3.7 Testes de Kerberos e DNS

```
1 kinit administrator@CORP.LOGAMTECH.LOCAL
2 klist
3 host -t A dcl.corp.logamtech.local
```

- **kinit**: autentica e obtém ticket Kerberos.
- **klist**: exibe ticket válido.
- **host**: valida resolução DNS interna.

3.8 Criação de Grupos e Usuários do Domínio

```
1 sudo samba-tool group add "Administradores_Logam" --description="Grupo com
privilegios administrativos no domínio"
```

```
2 sudo samba-tool group add "Users_Logam" --description="Grupo padr o de  
3   usu rios do dom nio"  
4 sudo samba-tool user create andre 'Andre!2025'  
5 sudo samba-tool user create renata 'Renata!2025'  
6 sudo samba-tool group addmembers "Administradores_Logam" Martha Gustavo  
   Patricia Ricardo Nathalia Andre
```

3.9 Ingresso de EC2 no Domínio

3.9.1 Criar a Instância EC2 do Cliente

- Nome: client-01
- Sistema Operacional: Ubuntu Server 22.04 LTS
- Tipo de Instância: t3.large
- Security Group: group-sg-client
- Elastic IP: 98.90.28.104

3.9.2 Atualizar pacotes do sistema

```
1 sudo apt update && sudo apt upgrade -y
```

3.9.3 Instalar dependências de comunicação com o AD

```
1 sudo apt install realmd sssd-ad sssd-tools adcli krb5-user samba-common -y
```

3.9.4 Verificar descoberta do domínio

```
1 realm discover corp.logamtech.local
```

3.9.5 Corrigir DNS (caso necessário)

```
1 sudo nano /etc/resolv.conf
2 nameserver 10.0.1.162
3 search corp.logamtech.local
```

3.9.6 Ingressar cliente no domínio

```
1 sudo realm join --user=administrator@CORP.LOGAMTECH.LOCAL corp.logamtech.
    local
```

3.9.7 Validar se o EC2 foi vinculado corretamente ao domínio

```
1 realm list
```

3.9.8 Validar autenticação de um usuário do domínio

```
1 id martha@corp.logamtech.local
2 getent passwd martha@corp.logamtech.local
```

3.10 Criar a Instância EC2 para Gerenciamento de GPOs

- Nome: win-server-gpo
- Sistema Operacional: Microsoft Windows Server 2019 Base
- Tipo de Instância: t3.large
- Security Group: group-sg-client
- Elastic IP: 52.202.113.69

3.10.1 Configurar Placa de Rede

3.11 Ingressar o Windows Server no Domínio Samba/AD

- Domínio: corp.logamtech.local

Tabela 5 – Configuração de IP e DNS — Windows Server

Campo	Valor	Descrição
IP address	10.0.1.122	IP privado fixo da instância
Subnet mask	255.255.255.0	Máscara da sub-rede
Default gateway	10.0.1.1	Gateway interno da VPC
Preferred DNS server	10.0.1.162	IP do DC Linux (dc1-puc)
Alternate DNS server	*em branco*	DNS de fallback

- Controlador principal: dc1-puc (Samba4 no Linux)
- Função: Administração de GPOs e gerenciamento gráfico do domínio.

3.11.1 Acessar configurações de Domínio

1 SystemPropertiesComputerName

- a) Clique em **Change...**
- b) Selecione **Domain**
- c) Digite: corp.logamtech.local

3.11.2 Inserir credenciais do domínio

1 Username: administrator@CORP.LOGAMTECH.LOCAL
2 Password: [senha definida durante o provisionamento]

3.11.3 Resultado esperado

1 Welcome to the corp.logamtech.local domain

3.11.4 Verificar Ingresso no Domínio

1 systeminfo | findstr /B /C:"Domain"

Resultado esperado:

1 Domain: corp.logamtech.local

3.11.5 Testar Autenticação Kerberos

```
1 klist
```

- A presença de um ticket válido confirma a autenticação Kerberos.

3.12 Instalar Ferramentas de Administração de GPO (RSAT)

3.12.1 Instalar via PowerShell

```
1 Install-WindowsFeature -Name RSAT-AD-Tools, RSAT-AD-PowerShell, RSAT-DNS-  
    Server, GPMC
```

3.12.2 Verificar Instalação

```
1 # Abrir GPMC  
2 gpmc.msc  
3  
4 # Abrir AD  
5 dsa.msc  
6  
7 # Abrir DNS  
8 dnsmgmt.msc
```

3.13 Criação e Gerenciamento de GPOs (Group Policy Objects)

3.13.1 Criar GPO — “Bloquear Troca de Papel de Parede”

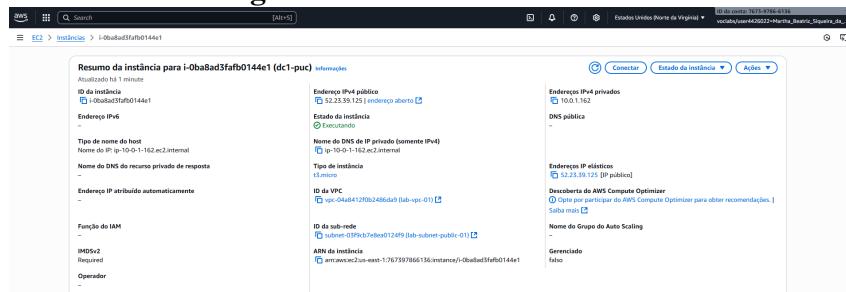
- Objetivo: Impedir que usuários alterem o plano de fundo do desktop, definindo um papel de parede fixo.

```
1 New-GPO -Name "Bloquear Troca de Papel de Parede" -Comment "Impede
2     altera o do plano de fundo do desktop"
3 New-GPLink -Name "Bloquear Troca de Papel de Parede" -Target "DC=corp,DC=
4     logamtech,DC=local"
5
6 Set-GPRegistryValue -Name "Bloquear Troca de Papel de Parede"
7     -Key "HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\ActiveDesktop"
8         "
9     -ValueName "NoChangingWallPaper" -Type DWord -Value 1
10
11 Set-GPRegistryValue -Name "Bloquear Troca de Papel de Parede"
12     -Key "HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System"
13     -ValueName "Wallpaper" -Type String -Value "C:\Windows\Web\Wallpaper\
14         Windows\img0.jpg"
15
16 gpupdate /force
```

3.13.2 Validar políticas aplicadas

```
1 Get-GPO -All
```

- O comando lista todas as GPOs criadas e vinculadas no domínio.

Figura 2 – Resumo da instância

Fonte: Elaborado pelos autores

A Figura 2 apresenta a instância EC2 utilizada para hospedar o servidor DNS, executando o sistema operacional **Ubuntu Server 24.04 LTS**, do tipo **t3.micro**, com IP privado **10.0.1.162** e Elastic IP público fixo. Foram instalados os serviços **Samba 4**, **Kerberos**, **Winbind** e **SMBClient**, responsáveis pela autenticação e resolução de nomes no domínio.

Figura 3 – Regras de Segurança do Servidor DNS (Security Group)

Regras de entrada

Nome	ID da regra do grupo de se...	Intervalo de po...	Protocolo	Origem	Grupos de segurança	Descrição
sgr-0062f2eef543d0f2	464	UDP	10.0.0.0/16	group-sq-ad	-	
sgr-06129e435a5f654aa	389	TCP	10.0.0.0/16	group-sq-ad	-	
sgr-06f5ef727a7a1ad5	88	TCP	10.0.0.0/16	group-sq-ad	-	
sgr-081c856d410ba625	1024-65535	TCP	10.0.0.0/16	group-sq-ad	-	
sgr-01bde1ee1a88e103	53	TCP	0.0.0.0/0	group-sq-ad	-	
sgr-0060ed2523db4e0f44	22	TCP	0.0.0.0/0	group-sq-ad	-	
sgr-0729750e0bf3c3e8	88	UDP	10.0.0.0/16	group-sq-ad	-	
sgr-079489e757eb516f	135	TCP	10.0.0.0/16	group-sq-ad	-	
sgr-090e044d9b550d97	464	TCP	10.0.0.0/16	group-sq-ad	-	
sgr-0db232216278ab9w	389	UDP	10.0.0.0/16	group-sq-ad	-	

Regras de saída

Nome	ID da regra do grupo de se...	Intervalo de po...	Protocolo	Destino	Grupos de segurança	Descrição
sgr-0800cf2a794a59123	Todos	Todos	0.0.0.0/0	group-sq-ad	-	

Fonte: Elaborado pelos autores

A Figura 3 mostra o **Security Group** configurado para o servidor DNS. As regras de entrada e saída permitem comunicação nas portas 53 (DNS), 88 (Kerberos), 389 (LDAP) e 445 (SMB), garantindo o funcionamento correto da autenticação e resolução de nomes dentro da VPC.

Figura 4 – Conexão via SSH com a instância EC2

```
ubuntu@elc:~          x + v
PS C:\Users\marthSSH Chaves> ssh -l "Martha_Ubuntu_01.pem" ubuntu@52.23.39.125
Welcome to Ubuntu 20.04 LTS (GNU/Linux 6.24.0-1014-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System Information as of Sun Oct 19 21:48:11 UTC 2025

System load: 0.0          Temperature:      -273.1 C
Usage of /: 39.3% of 6.71GB Processes:        167
Memory usage: 48%          Users Logged in:   0
Swap usage:  0%          IPv6 address for ens5: 10.0.1.162

* Ubuntu Pro delivers the most comprehensive open source security and
  compliance features.

https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

13 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Sat Oct 18 23:55:52 2025 from 191.35.60.34
ubuntu@elc:~$ |
```

Fonte: Elaborado pelos autores

A Figura 4 demonstra a conexão SSH realizada via PowerShell, utilizando a chave privada `Martha_Ubuntu_01.pem` para autenticação segura do usuário `ubuntu`. Essa etapa garante acesso remoto e controle administrativo total sobre a instância EC2.

Figura 5 – Instalação dos pacotes

```

# netsh winhttp /F :5 dgbr -l | grep -- "samba\winbind\smbclient\ldbh\ntp"
kb5-config          2.7          all      Configuration files for Kerberos Version 5
krb5-libs            1.10         all      integrated Kerberos libraries for Windows
krb5-user           1.20         amd64   basic programs to authenticate using MIT Kerberos
libgsapi-krb5-2:amd64 1.20         lib64   MIT Kerberos runtime libraries - GSAPI Mechanism
libgsapi-krb5-2:arm64 1.20         arm64   MIT Kerberos runtime libraries - GSAPI Mechanism
libkrb5support-0:amd64 1.20         lib64   MIT Kerberos runtime libraries - Support library
libkrb5support-0:arm64 1.20         arm64   MIT Kerberos runtime libraries - Support library
libldb-2:amd64        2.8          amd64   LDAP-like embedded database - shared library
libldb-2:arm64         2.8          arm64   Samba named pipe authentication plugin
libldb-3:amd64         2.9          amd64   Windows domain authentication plugin
libldb-3:arm64         2.9          arm64   Samba winbind client library
python3-3:ldb          2.9          amd64   Python 3 bindings for libldb
samba-19.3-0:samba     2.4.19       amd64   Python 3 bindings for Samba
samba-19.3-0:samba     2.4.19       lib64   Samba/CIFS file, print, and login server for Unix
samba-19.3-0:samba     2.4.19       lib64   Samba control files to run a Domain Controller
samba-common           2.4.19       all      Samba common files used by both the Samba server and client
samba-common-bin        2.4.19       lib64   Samba common files used by both the server and the client
samba-common-modules:amd64 2.4.19       amd64   Samba common modules Database
samba-common-modules:arm64 2.4.19       arm64   Samba common modules Database
samba-libs:amd64        2.4.19       amd64   Samba core libraries
samba-libs:arm64        2.4.19       arm64   Samba core libraries
samba-vfs-modules:amd64 2.4.19       amd64   Samba Virtual Filesystem plugins
samba-vfs-modules:arm64 2.4.19       arm64   service to resolve user and group information from Windows NT se

```

Fonte: Elaborado pelos autores

A Figura 5 apresenta a instalação dos pacotes necessários ao ambiente do domínio, como samba, krb5-config, winbind, smbclient, dnsutils e ntp. Esses pacotes são fundamentais para o funcionamento do **Active Directory** e do servidor DNS interno.

Figura 6 – Verificação do serviço Samba AD

```
● samba-ad-dc.service - Samba AD Daemon
   Loaded: loaded (/usr/lib/systemd/system/samba-ad-dc.service; enabled; preset: enabled)
   Active: active (running) since Sun 2025-10-19 18:46:15 UTC; 3h 1min ago
     Docs: man:samba(8)
           man:samba(7)
           man:samba(6)
           man:samba(5)
 Processes: 405 Events: 0
Main PID: 599 (samba)
Status: "samba: ready to serve connections..."
   Memory: 1.0G (peak: 1601.3M)
Memory: 300.3M (peak: 447.3M)
   CPU: 47.241s
CGroup: /system.slice/samba-ad-dc.service
          └─ 599 /usr/sbin/smbd -D --optionserver role check:inhibit=yes --foreground
    765 * [samba] task[765] pre-fork master process(766)*
    766 * [samba] task[766] pre-fork master*
    767 * [samba] task[s3f5] pre-fork master*
    768 * [samba] task[768] pre-fork master process(769)*
    769 * [samba] task[769] pre-fork master*
    770 /usr/sbin/smbd -D --optionserver role check:inhibit=yes --foreground
    771 * [samba] tfork waiter process(772)*
    772 * [samba] tfork waiter process(773)*
    773 * [samba] tfork waiter process(777)*
    774 * [samba] tfork waiter process(778)*
    775 * [samba] task[mpc] pre-fork worker(0)*
    776 * [samba] task[mpc] pre-fork master*
    777 * [samba] tfork waiter process(781)*
    778 * [samba] tfork waiter process(782)*
    779 * [samba] tfork waiter process(783)*
    780 * [samba] task[ldan] pre-fork master*
    781 * [samba] task[ldan] pre-fork worker(0)*
    782 * [samba] task[ldan] pre-fork worker(1)*
    783 * [samba] task[rpc] pre-fork worker(1)*
    784 * [samba] task[rpc] pre-fork master*
    785 * [samba] tfork waiter process(787)*
    786 * [samba] task[mpc] pre-fork master(2)*
    787 * [samba] task[mpc] pre-fork worker(2)*
    788 * [samba] tfork waiter process(798)*
    789 * [samba] tfork waiter process(793)*
    790 * [samba] task[ldan] pre-fork master*
    791 * [samba] task[ldan] pre-fork worker(0)*
    792 * [samba] tfork waiter process(795)*
```

Fonte: Elaborado pelos autores

A Figura 6 exibe o resultado do comando `systemctl status samba-ad-dc`, confirmando que o serviço **Samba AD/DC** está ativo e configurado para iniciar automaticamente, garantindo o funcionamento contínuo do Controlador de Domínio.

Figura 7 – Exibição do nível funcional do domínio

```
ubuntu@dc1:~$ sudo samba-tool domain level show
Domain and forest function level for domain 'DC=corp,DC=logamtech,DC=local'

Forest function level: (Windows) 2008 R2
Domain function level: (Windows) 2008 R2
Lowest function level of a DC: (Windows) 2008 R2
ubuntu@dc1:~$ |
```

Fonte: Elaborado pelos autores

A Figura 7 apresenta a execução do comando `samba-tool domain level show`, que exibe os níveis funcionais *forest* e *domain*. Essa verificação confirma que o domínio foi promovido corretamente e está funcional no ambiente.

Figura 8 – Resolução de DNS

```
ubuntu@dc1:~$ host -t A dc1.corp.logamtech.local
dc1.corp.logamtech.local has address 10.0.1.162
ubuntu@dc1:~$ |
```

Fonte: Elaborado pelos autores

A Figura 8 mostra o teste de resolução de nomes realizado com o comando `host -t A dc1.corp.logamtech.local`. O retorno confirma que o **DNS interno** está resolvendo corretamente o endereço IP da instância do controlador de domínio.

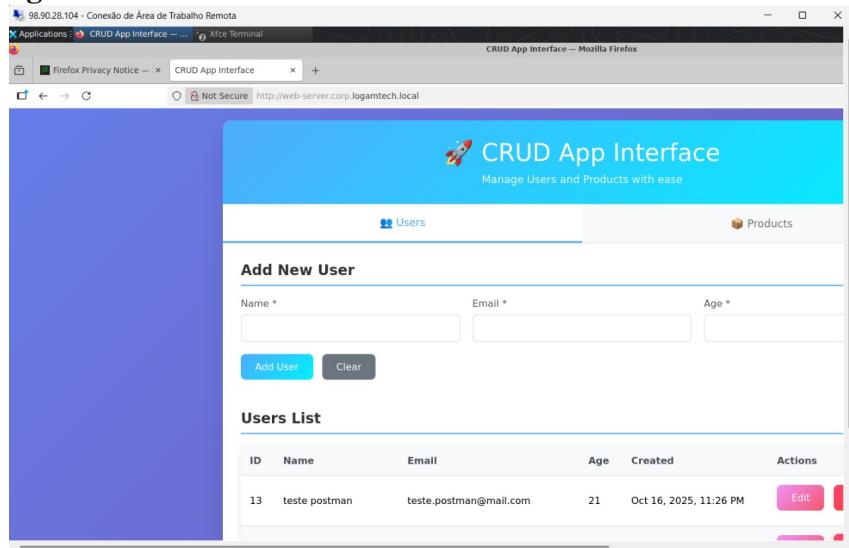
Figura 9 – Teste de resolução de subdomínios FTP e Web

```
ubuntu@dc1:~$ host ftp.corp.logamtech.local
host web-server.corp.logamtech.local
ftp.corp.logamtech.local has address 18.212.95.192
web-server.corp.logamtech.local has address 54.145.137.231
ubuntu@dc1:~$ |
```

Fonte: Elaborado pelos autores

A Figura 9 apresenta a validação dos registros de subdomínios configurados no **Route 53** — `ftp.corp.logamtech.local` e `web-server.corp.logamtech.local`. O teste comprova que a resolução interna entre instâncias ocorre conforme o esperado.

Figura 10 – Acessando o CRUD do subdomínio do Servidor Web



Fonte: Elaborado pelos autores

A Figura 10 mostra o acesso ao **CRUD do servidor Web** por meio do subdomínio `web-server.corp.logamtech.local`. O teste foi realizado no navegador Firefox a partir de um cliente já ingressado no domínio, comprovando a resolução e o acesso via DNS.

Figura 11 – Teste de autenticação Kerberos com o administrador

```
ubuntu@dc1:~$ kinit administrator@CORP.LOGAMTECH.LOCAL
Password for administrator@CORP.LOGAMTECH.LOCAL:
Warning: Your password will expire in 26 days on Sat Nov 15 15:44:25 2025
ubuntu@dc1:~$ |
```

Fonte: Elaborado pelos autores

A Figura 11 apresenta o comando `kinit administrator@CORP.LOGAMTECH.LOCAL`, utilizado para autenticar o administrador do domínio e gerar o ticket Kerberos. A resposta positiva confirma que o serviço de autenticação Kerberos está operacional.

Figura 12 – Listagem do ticket Kerberos ativo

```
ubuntu@dc1:~$ klist
Ticket cache: FILE:/tmp/krb5cc_1000
Default principal: administrator@CORP.LOGAMTECH.LOCAL

Valid starting     Expires            Service principal
10/19/25 22:14:17  10/20/25 08:14:17  krbtgt/CORP.LOGAMTECH.LOCAL@CORP.LOGAMTECH.LOCAL
                  renew until 10/20/25 22:14:10
ubuntu@dc1:~$ |
```

Fonte: Elaborado pelos autores

A Figura 12 mostra o comando `klist`, que lista o ticket Kerberos ativo emitido para o administrador, exibindo as informações de validade e confirmação da autenticação no domínio.

Figura 13 – Listagem de usuários criados no domínio

```
ubuntu@dc1:~$ sudo samba-tool user list
carla
thiago
renata
Omar
laura
Administrator
gustavo
diego
rodrigo
Gabriel
nathalia
rafael
pedro
patricia
bruno
caroline
Alice
martha
andre
elisa
cristina
paulo
ricardo
henrique
isabela
bruna
Guest
sofia
lorena
julia
Leandro
joao
krbtgt
daniel
arthur
vinicius
monique
fernanda
amanda
```

Fonte: Elaborado pelos autores

A Figura 13 apresenta a saída do comando `samba-tool user list`, que lista todos os usuários cadastrados no domínio CORP.LOGAMTECH.LOCAL. O resultado confirma que as contas foram criadas corretamente e reconhecidas pelo serviço LDAP do Samba AD.

Figura 14 – Verificação de grupos criados no domínio

```
ubuntu@dc1:~$ sudo samba-tool group list
Guests
Incoming Forest Trust Builders
Enterprise Read-only Domain Controllers
Windows Authorization Access Group
Protected Users
Account Operators
Cert Publishers
Network Configuration Operators
Server Operators
Domain Controllers
Denied RODC Password Replication Group
DnsAdmins
RAS and IAS Servers
DnsUpdateProxy
Users_Logam
Distributed COM Users
Domain Admins
Allowed RODC Password Replication Group
Enterprise Admins
Event Log Readers
Replicator
Group Policy Creator Owners
Print Operators
Domain Users
Performance Log Users
Schema Admins
Terminal Server License Servers
Domain Computers
Certificate Service DCOM Access
IIS_IUSRS
Domain Guests
Read-only Domain Controllers
Administrators
Remote Desktop Users
Administradores_Logam
Performance Monitor Users
Pre-Windows 2000 Compatible Access
Backup Operators
Users
```

Fonte: Elaborado pelos autores

A Figura 14 apresenta o resultado do comando `samba-tool group list`, que lista todos os grupos criados no domínio, como *Users_Logam* e *Administradores_Logam*. O teste confirma que os grupos foram criados corretamente e reconhecidos pelo serviço LDAP do Samba AD.

Figura 15 – Autenticação Kerberos com usuário comum

```
ubuntu@dc1:~$ kinit caroline@CORP.LOGAMTECH.LOCAL
Password for caroline@CORP.LOGAMTECH.LOCAL:
Warning: Your password will expire in 40 days on Sat Nov 29 11:45:17 2025
ubuntu@dc1:~$ |
```

Fonte: Elaborado pelos autores

Na Figura 15, foi executado o comando `kinit caroline@CORP.LOGAMTECH.LOCAL`, testando a autenticação Kerberos com uma conta de usuário comum. O retorno confirma que o domínio emite corretamente tickets para usuários padrão, além dos administrativos.

Figura 16 – Ingresso do cliente EC2 Linux no domínio

```
ubuntu@ip-10-0-1-64:~$ realm list
corp.logamtech.local
  type: kerberos
  realm-name: CORP.LOGAMTECH.LOCAL
  domain-name: corp.logamtech.local
  configured: kerberos-member
  server-software: active-directory
  client-software: sssd
  required-package: sssd-tools
  required-package: sssd
  required-package: libnss-sss
  required-package: libpam-sss
  required-package: adcli
  required-package: samba-common-bin
  login-formats: %U@corp.logamtech.local
  login-policy: allow-realm-logins
ubuntu@ip-10-0-1-64:~$ |
```

Fonte: Elaborado pelos autores

A Figura 16 apresenta a execução do comando `realm list`, confirmando que a instância Linux foi associada corretamente ao domínio `corp.logamtech.local`. Esse procedimento valida a integração entre o cliente e o Controlador de Domínio Samba/AD.

Figura 17 – Validação de autenticação de usuário no cliente

```
ubuntu@ip-10-0-1-64:~$ id martha@corp.logamtech.local
uid=9896001103(martha@corp.logamtech.local) gid=9896008513(domain users@corp.logamtech.local) groups=9896008513(domain users@corp.logamtech.local), 9896008512(domain admins@corp.logamtech.local), 9896008572(denied rodc password replication group@corp.logamtech.local)
ubuntu@ip-10-0-1-64:~$ |
```

Fonte: Elaborado pelos autores

A Figura 17 mostra o comando `id caroline@corp.logamtech.local`, exibindo o UID e os grupos aos quais o usuário pertence. O resultado confirma que a autenticação está sendo feita corretamente pelo domínio e o cliente reconhece as credenciais de rede.

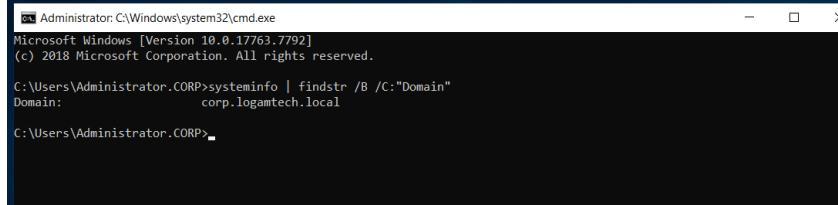
Figura 18 – Teste de comunicação entre cliente e DC

```
ubuntu@ip-10-0-1-64:~$ ping dc1.corp.logamtech.local
PING dc1.corp.logamtech.local (10.0.1.162) 56(84) bytes of data.
64 bytes from 10.0.1.162: icmp_seq=1 ttl=64 time=0.261 ms
64 bytes from 10.0.1.162: icmp_seq=2 ttl=64 time=0.221 ms
64 bytes from 10.0.1.162: icmp_seq=3 ttl=64 time=0.253 ms
64 bytes from 10.0.1.162: icmp_seq=4 ttl=64 time=0.248 ms
64 bytes from 10.0.1.162: icmp_seq=5 ttl=64 time=0.238 ms
64 bytes from 10.0.1.162: icmp_seq=6 ttl=64 time=0.308 ms
64 bytes from 10.0.1.162: icmp_seq=7 ttl=64 time=0.238 ms
64 bytes from 10.0.1.162: icmp_seq=8 ttl=64 time=0.276 ms
64 bytes from 10.0.1.162: icmp_seq=9 ttl=64 time=0.238 ms
64 bytes from 10.0.1.162: icmp_seq=10 ttl=64 time=0.233 ms
64 bytes from 10.0.1.162: icmp_seq=11 ttl=64 time=0.230 ms
64 bytes from 10.0.1.162: icmp_seq=12 ttl=64 time=0.234 ms
^C
--- dc1.corp.logamtech.local ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11263ms
rtt min/avg/max/mdev = 0.221/0.248/0.308/0.023 ms
```

Fonte: Elaborado pelos autores

A Figura 18 apresenta a execução do comando `ping dc1.corp.logamtech.local` a partir do cliente ingressado no domínio. O retorno positivo confirma a comunicação ICMP com o Controlador de Domínio e a correta resolução de nomes via DNS interno.

Figura 19 – Ingresso da instância Windows Server (win-server-gpo) no domínio



```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.17763.7792]
(c) 2018 Microsoft Corporation. All rights reserved.

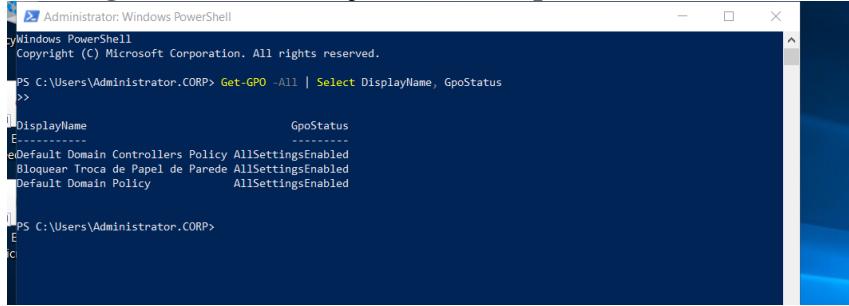
C:\Users\Administrator.CORP>systeminfo | findstr /B /C:"Domain"
Domain:          corp.logamtech.local

C:\Users\Administrator.CORP>
```

Fonte: Elaborado pelos autores

A Figura 19 mostra o comando `systeminfo`, evidenciando que o servidor *win-server-gpo* foi adicionado com sucesso ao domínio `corp.logamtech.local`. Esse servidor foi utilizado para o gerenciamento de políticas de grupo (GPO).

Figura 20 – Verificação da GPO aplicada no domínio



```
Administrator: Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator.CORP> Get-GPO -All | Select DisplayName, GpoStatus
>>
DisplayName          GpoStatus
E-----
eDefault Domain Controllers Policy AllSettingsEnabled
Bloquear Troca de Papel de Parede AllSettingsEnabled
Default Domain Policy   AllSettingsEnabled

PS C:\Users\Administrator.CORP>
E
G
```

Fonte: Elaborado pelos autores

Por fim, a Figura 20 apresenta a verificação da política de grupo aplicada no domínio, confirmando que a GPO “**Bloquear troca de papel de parede**” foi replicada e aplicada com sucesso nos clientes ingressados.

3.14 Servidor FTP

O servidor FTP (File Transfer Protocol) foi configurado na AWS com o objetivo de permitir transferência segura de arquivos entre a sede e as filiais. O protocolo FTPS (FTP Secure) foi adotado, utilizando criptografia TLS/SSL para proteger os dados durante o envio e recebimento.

Tabela 6 – Informações do servidor (AWS)

Nome do Servidor	Endereço IP Público	Endereço IP Privado	Usuário de Acesso	Função
novoftp	54.160.164.64	172.31.17.229	N/A	Permitir transferência segura de arquivos entre a sede e as filiais.

Fonte: Criado pelos autores

Principais configurações realizadas:

3.15 Criar o EC2 para o Servidor FTP

- Nome: novoftp
- Sistema operacional: Ubuntu 24.04 LTS
- Tipo de instância: t3.micro
- Security Group: launch-wizard-4
- Storage: 8GB

3.16 Configuração de Segurança (Security Groups)

3.16.1 Regras de Entrada — *launch-wizard-4*

- Descrição: As regras de segurança no FTP (ou sua falta) determinam como o acesso é controlado e se a transferência de dados é criptografada.
- Objetivo: Proteger contra acesso não autorizado, criptografar os dados para prevenir interceptação e controlar quem pode fazer o quê, com base na autenticação e na autorização.
10.0.0.0/16.

Tabela 7 – Regras de entrada — launch-wizard-4

Tipo	Protocolo	Porta / Intervalo	Origem	Descrição
TCP personalizado	TCP	20	0.0.0.0/0	N/A
TCP personalizado	TCP	21	0.0.0.0/0	N/A
TCP personalizado	TCP	12.000	0.0.0.0/0	N/A
TCP personalizado	TCP	12.100	0.0.0.0/0	N/A
SSH	TCP	22	0.0.0.0/0	Acesso remoto

3.17 Configuração do Servidor FTP

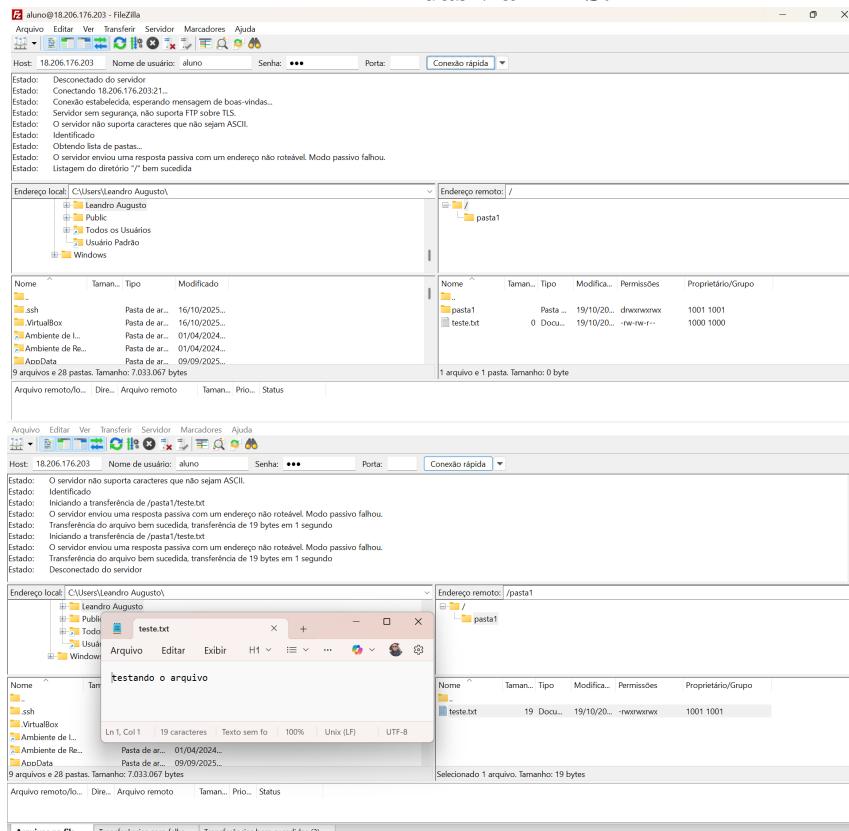
3.17.1 Acesso à instância

```
1 ssh -i "novoftp.pem" ubuntu@ec2-54-160-164-64.compute-1.amazonaws.com
```

3.17.2 Atualização dos pacotes

```
1 sudo apt update && sudo apt upgrade -y
```

Figura 21 – Inserir print do painel do servidor FTP, pastas criadas e conexões autenticadas via FTPS.



Fonte: Elaborado pelos autores

3.18 Servidor VPN (OpenVPN)

O serviço de VPN (Virtual Private Network) foi implementado utilizando o OpenVPN, hospedado na AWS. Sua função é garantir uma comunicação segura e criptografada entre os colaboradores remotos e os sistemas corporativos.

Tabela 8 – Informações do servidor (AWS)

Nome do Servidor	Endereço IP Público	Endereço IP Privado	Usuário de Acesso	Função
logamTech	54.234.104.6	172.31.29.60	N/A	Conectar a VPN

Fonte: Criado pelos autores

Principais configurações realizadas:

3.19 Criar a Instância EC2 para o Servidor VPN

- Nome: vpn-server
- Sistema operacional: Ubuntu 24.04 LTS
- Tipo de instância: t3.micro
- Storage: 8 GB
- Security Group: VPN
- IP privado: 172.31.17.170

3.20 Configurar Security Group da VPN

- Nome: VPN
- Descrição: Security Group para servidor VPN
- Regras de entrada:
 - UDP 1194 – Acesso de clientes OpenVPN: 0.0.0.0/0
 - SSH 22 – Acesso administrativo: 0.0.0.0/0
 - ICMP (Echo Request) – Permitir testes de conectividade (ping): 0.0.0.0/0
- Regras de saída:
 - Todos: 0.0.0.0/0 : 0

3.21 Instalar e Configurar o OpenVPN

3.21.1 Acessar a instância EC2 *vpn-server*

```
1 ssh -i "vpn-chave.pem" ubuntu@54.210.126.47
```

3.21.2 Atualizar o sistema

```
1 sudo apt update && sudo apt upgrade -y
```

3.21.3 Instalar o OpenVPN

```
1 sudo apt install openvpn -y
```

3.22 Gerar e Enviar o Arquivo de Configuração do Cliente (.ovpn)

```
1 wget https://git.io/vpn -O openvpn-install.sh && bash openvpn-install.sh
```

3.22.1 Baixar o arquivo .ovpn para o computador

```
1 scp -i "vpn-chave.pem" ubuntu@54.210.126.47:/home/ubuntu/vpn_client_1.ovpn  
.
```

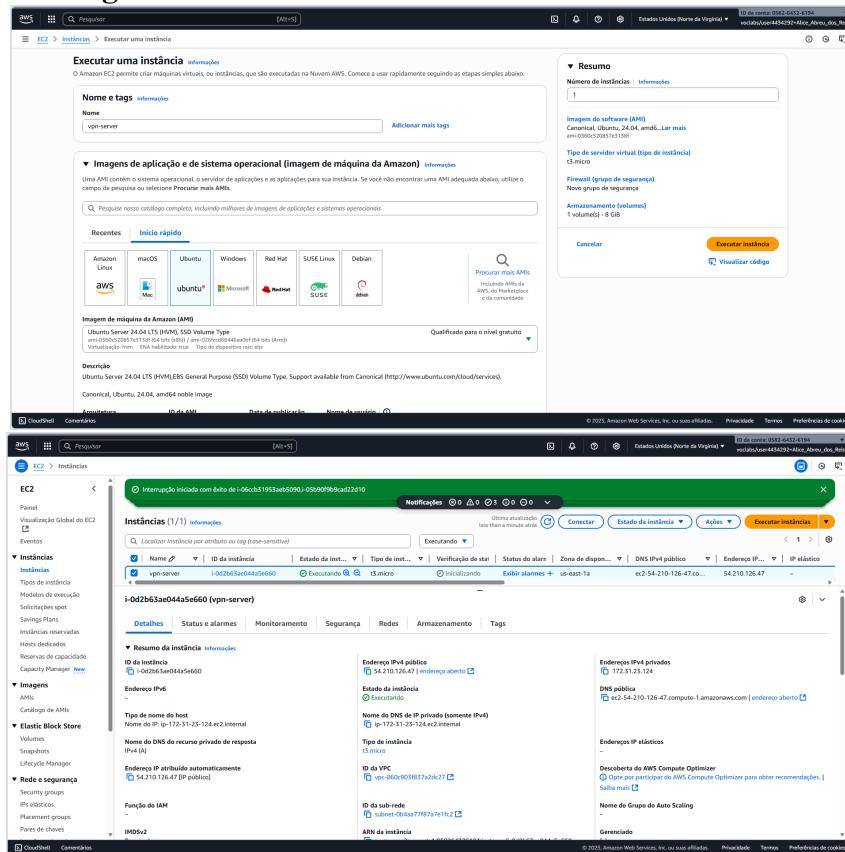
3.23 Conexão e Testes

3.23.1 Conectar ao servidor VPN no cliente (Ubuntu)

```
1 sudo apt install openvpn -y  
2 sudo openvpn --config ~/Downloads/vpn_client_1.ovpn
```

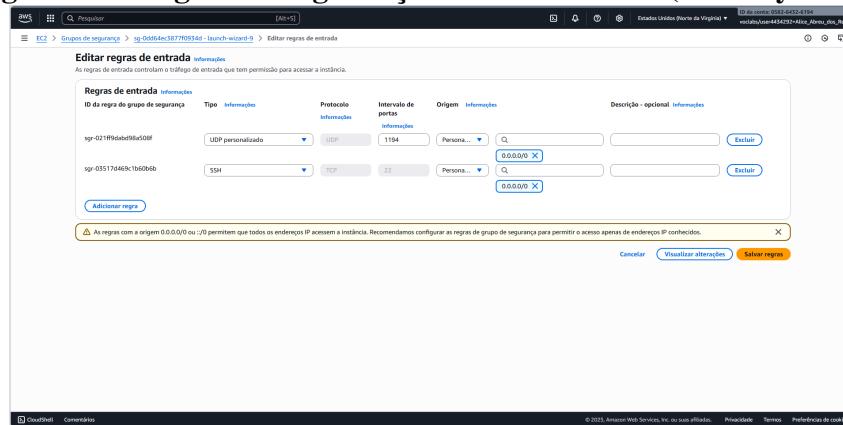
3.23.2 Testar conectividade com a rede privada

```
| ping 172.31.17.170
```

Figura 22 – Painel da Instância EC2 da FTP na AWS**Fonte: Elaborado pelos autores**

A Figura 22 apresenta a instância EC2 criada na AWS.

Figura 23 – Regras de Segurança do Servidor FTP (Security Group)



Fonte: Elaborado pelos autores

A Figura 23 mostra o Security Group configurado para a instância VPN.

Figura 24 – Conexão via SSH com a instância EC2

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

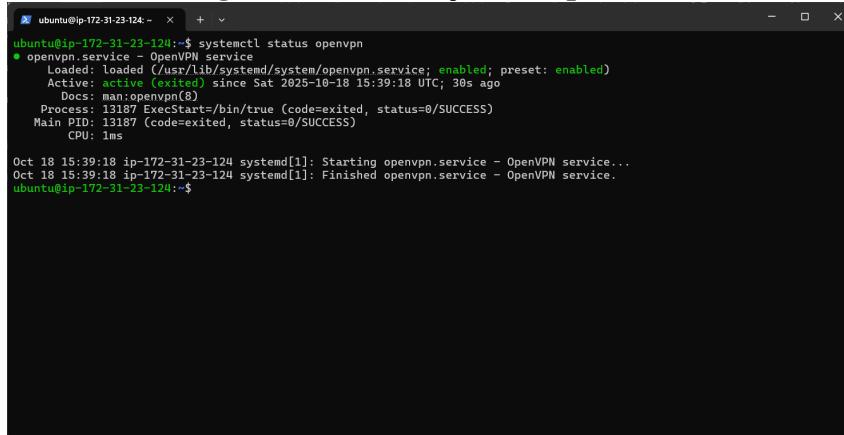
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\alice> ssh -i "vpn-chave.pem" ubuntu@ec2-54-210-126-47.compute-1.amazonaws.com
The authenticity of host 'ec2-54-210-126-47.compute-1.amazonaws.com (54.210.126.47)' can't be established.
ED25519 key fingerprint is SHA256:UbD18Cj0wQa5ZHW5zR8h1hcdVklpJ+268Hoop26UzeU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

Fonte: Elaborado pelos autores

A Figura 24 mostra a conexão SSH sendo estabelecida com a instância EC2.

Figura 25 – Instalação do OpenVPN



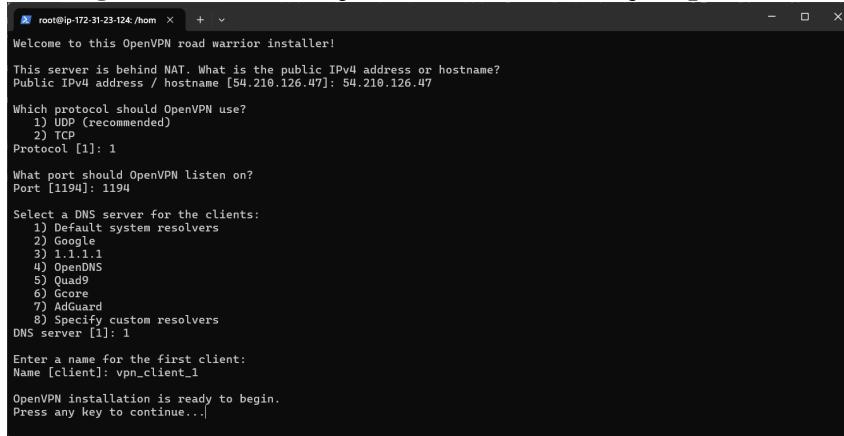
```
ubuntu@ip-172-31-23-124:~$ systemctl status openvpn
● openvpn.service - OpenVPN service
  Loaded: loaded (/usr/lib/systemd/system/openvpn.service; enabled; preset: enabled)
  Active: active (running) since Sat 2025-10-18 15:39:18 UTC; 30s ago
    Docs: man:openvpn(8)
   Process: 13187 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
    Main PID: 13187 (code=exited, status=0/SUCCESS)
      CPU: 1ms

Oct 18 15:39:18 ip-172-31-23-124 systemd[1]: Starting openvpn.service - OpenVPN service...
Oct 18 15:39:18 ip-172-31-23-124 systemd[1]: Finished openvpn.service - OpenVPN service.
ubuntu@ip-172-31-23-124:~$
```

Fonte: Elaborado pelos autores

Na Figura 25, é apresentada a etapa de instalação e configuração inicial do serviço OpenVPN no servidor Ubuntu hospedado na AWS.

Figura 26 – Verificação do status do serviço OpenVPN



```
Welcome to this OpenVPN road warrior installer!

This server is behind NAT. What is the public IPv4 address or hostname?
Public IPv4 address / hostname [54.210.126.47]: 54.210.126.47

Which protocol should OpenVPN use?
  1) UDP (recommended)
  2) TCP
Protocol [1]: 1

What port should OpenVPN listen on?
Port [1194]: 1194

Select a DNS server for the clients:
  1) Default system resolvers
  2) Google
  3) 1.1.1.1
  4) OpenDNS
  5) Quad9
  6) Gcore
  7) AdGuard
  8) Specify custom resolvers
DNS server [1]: 1

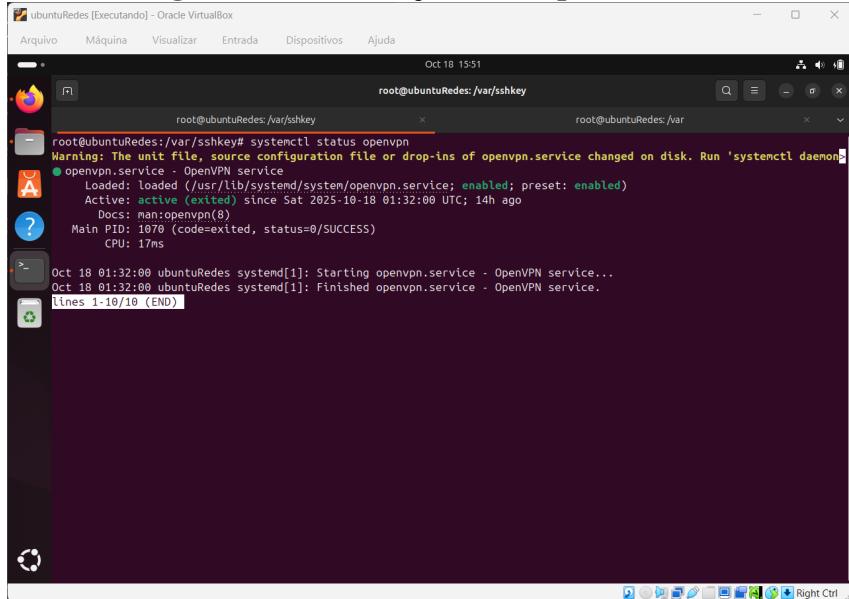
Enter a name for the first client:
Name [client]: vpn_client_1

OpenVPN installation is ready to begin.
Press any key to continue...|
```

Fonte: Elaborado pelos autores

A Figura 26 mostra o status do serviço OpenVPN na instância Ubuntu, indicando que ele está ativo e configurado para iniciar automaticamente com o sistema, pronto para aceitar conexões de clientes via VPN.

Figura 27 – Verificação na máquina virtual



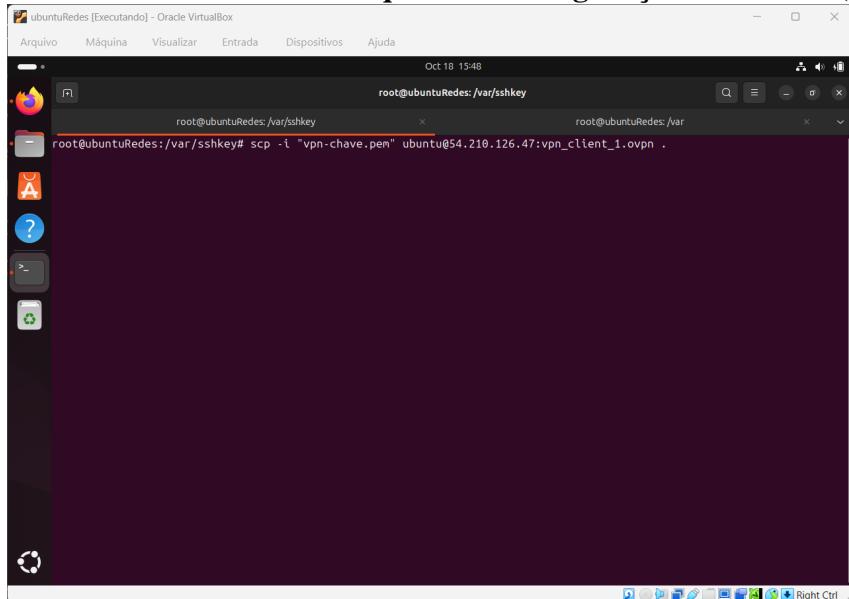
A screenshot of a terminal window titled "ubuntuRedes [Executando] - Oracle VirtualBox". The window has two tabs: "root@ubuntuRedes: /var/sshkey" and "root@ubuntuRedes: /var". The left tab shows the command "root@ubuntuRedes: /var/sshkey# systemctl status openvpn" and its output:

```
Oct 18 15:51 root@ubuntuRedes: /var/sshkey
root@ubuntuRedes: /var/sshkey# systemctl status openvpn
Warning: The unit file, source configuration file or drop-ins of openvpn.service changed on disk. Run 'systemctl daemon-reload' to reload units.
● openvpn.service - OpenVPN service
  Loaded: loaded (/usr/lib/systemd/system/openvpn.service; enabled; preset: enabled)
  Active: active (exited) since Sat 2025-10-18 01:32:00 UTC; 14h ago
    Docs: man:openvpn(8)
          Main PID: 1070 (code=exited, status=0/SUCCESS)
            CPU: 17ms
Oct 18 01:32:00 ubuntuRedes systemd[1]: Starting openvpn.service - OpenVPN service...
Oct 18 01:32:00 ubuntuRedes systemd[1]: Finished openvpn.service - OpenVPN service.
lines 1-10/10 (END)
```

Fonte: Elaborado pelos autores

A Figura 27 mostra novamente a verificação do status do serviço OpenVPN na máquina virtual.

Figura 28 – Transferência do Arquivo de Configuração do Cliente (.ovpn)



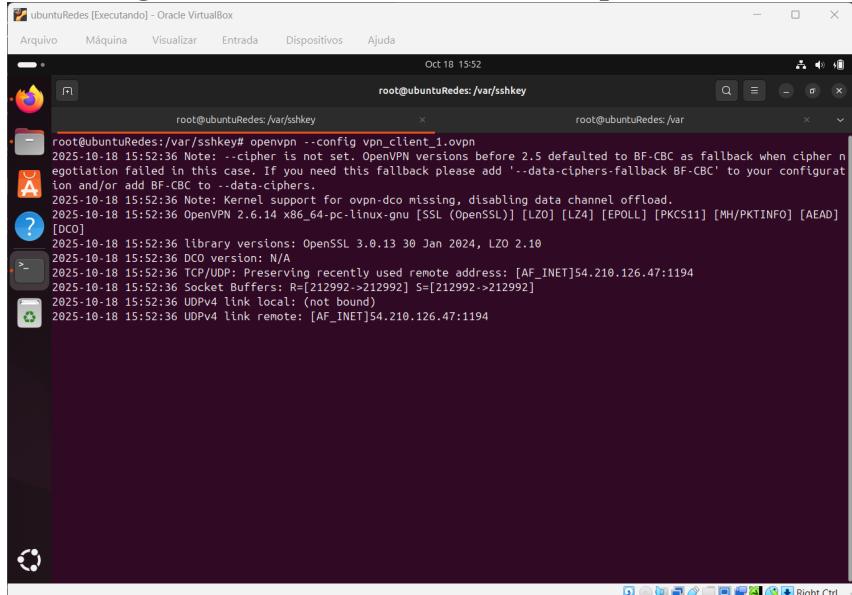
A screenshot of a terminal window titled "ubuntuRedes [Executando] - Oracle VirtualBox". The window has two tabs: "root@ubuntuRedes: /var/sshkey" and "root@ubuntuRedes: /var". The left tab shows the command "root@ubuntuRedes: /var/sshkey# scp -i "vpn-chave.pem" ubuntu@54.210.126.47:vpn_client_1.ovpn ." and its output:

```
Oct 18 15:48 root@ubuntuRedes: /var/sshkey
root@ubuntuRedes: /var/sshkey# scp -i "vpn-chave.pem" ubuntu@54.210.126.47:vpn_client_1.ovpn .
```

Fonte: Elaborado pelos autores

A Figura 28 mostra a cópia do arquivo de configuração do cliente do servidor EC2 para a máquina, contendo os dados necessários para a conexão VPN.

Figura 29 – Início da Conexão VPN pelo Cliente

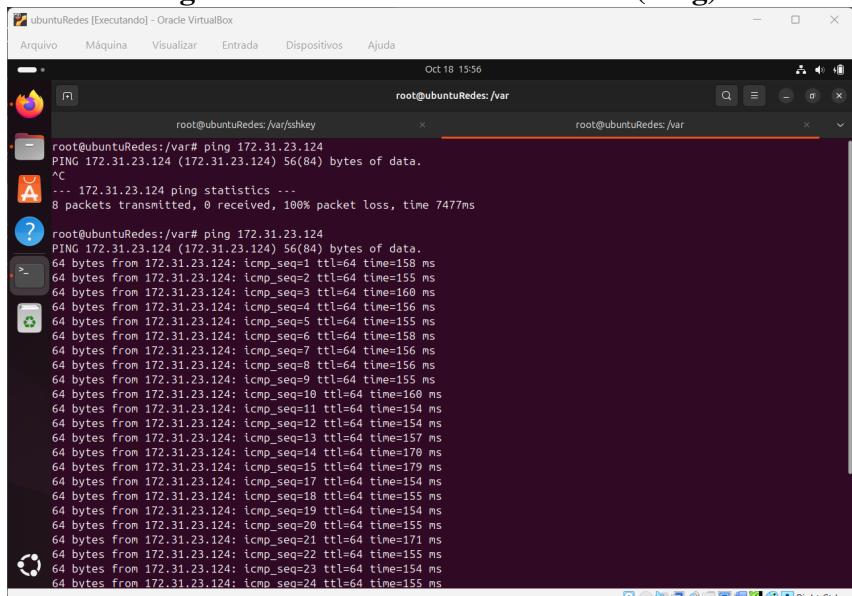


```
root@ubuntuRedes:~# openvpn --config vpn_client_1.ovpn
2025-10-18 15:52:36 Note: --cipher is not set. OpenVPN versions before 2.5 defaulted to BF-CBC as fallback when cipher negotiation failed in this case. If you need this fallback please add '-data-ciphers-fallback BF-CBC' to your configuration and/or add BF-CBC to --data-ciphers.
2025-10-18 15:52:36 Note: Kernel support for ovpn-dco missing, disabling data channel offload.
2025-10-18 15:52:36 OpenVPN 2.6.14 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD]
[DCO]
2025-10-18 15:52:36 library versions: OpenSSL 3.0.13 30 Jan 2024, LZO 2.10
2025-10-18 15:52:36 DCO version: N/A
2025-10-18 15:52:36 TCP/UDP: Preserving recently used remote address: [AF_INET]54.210.126.47:1194
2025-10-18 15:52:36 Socket Buffers: R=[212992->212992] S=[212992->212992]
2025-10-18 15:52:36 UDPv4 link local: (not bound)
2025-10-18 15:52:36 UDPv4 link remote: [AF_INET]54.210.126.47:1194
```

Fonte: Elaborado pelos autores

A Figura 29 mostra a execução do comando para iniciar a conexão VPN no cliente, estabelecendo o túnel seguro com o servidor.

Figura 30 – Teste de Conectividade (Ping)



```
root@ubuntuRedes:~# ping 172.31.23.124
PING 172.31.23.124 (172.31.23.124) 56(84) bytes of data.
^C
--- 172.31.23.124 ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7477ms

root@ubuntuRedes:~# ping 172.31.23.124
PING 172.31.23.124 (172.31.23.124) 56(84) bytes of data.
64 bytes from 172.31.23.124: icmp_seq=1 ttl=64 time=158 ms
64 bytes from 172.31.23.124: icmp_seq=2 ttl=64 time=155 ms
64 bytes from 172.31.23.124: icmp_seq=3 ttl=64 time=160 ms
64 bytes from 172.31.23.124: icmp_seq=4 ttl=64 time=156 ms
64 bytes from 172.31.23.124: icmp_seq=5 ttl=64 time=155 ms
64 bytes from 172.31.23.124: icmp_seq=6 ttl=64 time=158 ms
64 bytes from 172.31.23.124: icmp_seq=7 ttl=64 time=156 ms
64 bytes from 172.31.23.124: icmp_seq=8 ttl=64 time=156 ms
64 bytes from 172.31.23.124: icmp_seq=9 ttl=64 time=155 ms
64 bytes from 172.31.23.124: icmp_seq=10 ttl=64 time=160 ms
64 bytes from 172.31.23.124: icmp_seq=11 ttl=64 time=154 ms
64 bytes from 172.31.23.124: icmp_seq=12 ttl=64 time=154 ms
64 bytes from 172.31.23.124: icmp_seq=13 ttl=64 time=157 ms
64 bytes from 172.31.23.124: icmp_seq=14 ttl=64 time=178 ms
64 bytes from 172.31.23.124: icmp_seq=15 ttl=64 time=179 ms
64 bytes from 172.31.23.124: icmp_seq=16 ttl=64 time=154 ms
64 bytes from 172.31.23.124: icmp_seq=17 ttl=64 time=154 ms
64 bytes from 172.31.23.124: icmp_seq=18 ttl=64 time=155 ms
64 bytes from 172.31.23.124: icmp_seq=19 ttl=64 time=154 ms
64 bytes from 172.31.23.124: icmp_seq=20 ttl=64 time=155 ms
64 bytes from 172.31.23.124: icmp_seq=21 ttl=64 time=171 ms
64 bytes from 172.31.23.124: icmp_seq=22 ttl=64 time=155 ms
64 bytes from 172.31.23.124: icmp_seq=23 ttl=64 time=154 ms
64 bytes from 172.31.23.124: icmp_seq=24 ttl=64 time=155 ms
```

Fonte: Elaborado pelos autores

A Figura 30 apresenta a execução de um teste de conectividade ('ping') para verificar se a conexão VPN foi estabelecida corretamente. Esse passo confirma que o cliente consegue se comunicar com a rede remota através do túnel seguro.

3.24 Servidor DHCP

O serviço DHCP (Dynamic Host Configuration Protocol) foi configurado localmente no VirtualBox, com o objetivo de distribuir automaticamente endereços IP às máquinas da rede simulada. Essa configuração reduz falhas humanas e garante consistência no endereçamento de dispositivos.

Tabela 9 – Informações do servidor (VirtualBox)

Nome do Servidor	Endereço IP Público	Endereço IP Privado	Usuário de Acesso	Função
DebianSrv	x	192.168.1.1	x	Distribuição automática de endereços IP para clientes na rede interna.

Fonte: Criado pelos autores

Configurações do Servidor DHCP

3.4.1 Criação da Máquina Virtual

- Nome: DebianSrv
- Virtualizador: Oracle VM VirtualBox
- Sistema operacional: Debian 13
- Memória RAM: 1 GB
- Armazenamento: 10 GB
- Interfaces de rede:
 - **enp0s3**: modo NAT, usada para acesso à internet.
 - **enp0s8**: modo rede interna, utilizada para fornecer endereços IP aos clientes.

3.4.2 Configuração das Interfaces de Rede

No arquivo de configuração de rede `/etc/network/interfaces`, defina as interfaces:

```
1 # Interface conectada      rede externa (internet), recebe IP automaticamente
  via DHCP
2 auto enp0s3
3 iface enp0s3 inet dhcp
4
5 # Interface conectada      rede interna, usada como servidor DHCP, possui IP
  fixo
6 auto enp0s8
7 iface enp0s8 inet static
8   address 192.168.1.1
9   netmask 255.255.255.0
10  dns-nameservers 8.8.8.8 1.1.1.1
```

Para aplicar as alterações na configuração de rede, execute:

```
1 sudo systemctl restart networking
```

3.4.3 Instalação do Servidor DHCP

```
1 # Atualizar pacotes
2 sudo apt update -y && sudo apt upgrade -y
3
4 # Instalar isc-dhcp-server
5 sudo apt install isc-dhcp-server -y
```

3.4.4 Definição da Interface do DHCP

Ajuste a interface dhcp no arquivo /etc/default/isc-dhcp-server

```
1 INTERFACESv4="enp0s8"
```

3.4.5 Configuração do Escopo DHCP

O arquivo de configuração do DHCP localizado em /etc/dhcp/dhcpd.conf deve conter:

```
1 subnet 192.168.1.0 netmask 255.255.255.0 {
2   range 192.168.1.51 192.168.1.100;
3   option routers 192.168.1.1;
4   option domain-name-servers 8.8.8.8, 1.1.1.1;
5   option domain-name "exemple.org";
6 }
```

3.4.6 Reinicialização do Serviço e status

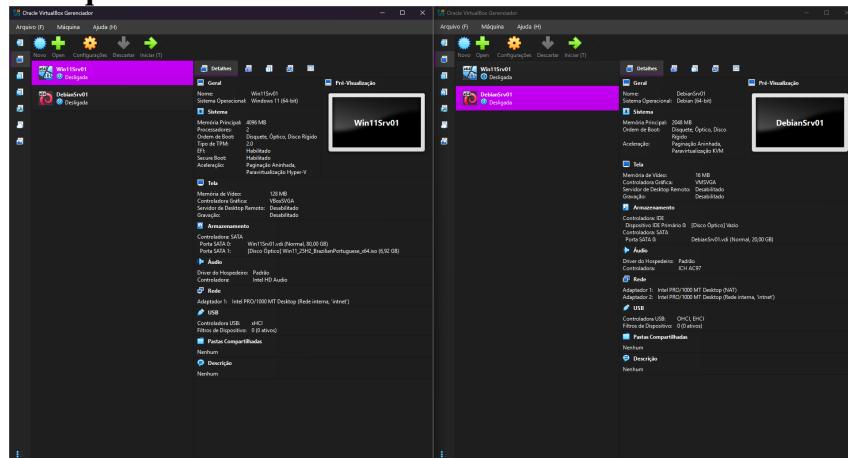
```
1 sudo systemctl restart isc-dhcp-server  
2 sudo systemctl status isc-dhcp-server
```

3.4.7 Testes de Conectividade

No cliente Windows, configure a interface de rede para obter o endereço IP automaticamente via DHCP e verifique o IP recebido usando:

```
1 ipconfig
```

Figura 31 – Máquina Virtual DebianSrv01 e Win11Srv01 no Oracle VM VirtualBox



Fonte: Elaborado pelos autores

A Figura 31 apresenta a máquina virtual criada para hospedar o servidor DHCP e o cliente que irá receber o IP. A VM Debian 13 possui duas interfaces de rede: enp0s3 (modo NAT) e enp0s8 (rede interna). A VM Windows 11 possui uma interface de rede interna para testes de recebimento de IP.

Figura 32 – Configuração das Interfaces de Rede

DebianDev01 [Executando] - Oracle VirtualBox

Arquivo Máquina Visualizar Entrada Dispositivos Ajuda

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# Interface com DHCP
auto ens3
iface ens3 inet dhcp

# Interface com IP fixo
auto ens3
iface ens3 inet static
    address 192.168.1.1
    netmask 255.255.255.0
    dns-nameservers 8.8.8.8 1.1.1.1

```

Fonte: Elaborado pelos autores

A Figura 32 mostra o conteúdo do arquivo /etc/network/interfaces, evidenciando que a interface enp0s3 recebe IP via DHCP e a enp0s8 possui IP fixo 192.168.1.1, atuando como gateway da rede interna.

Figura 33 – Verificação das Interfaces de Rede

```
DebianGrv1 [Ejecutando] - Oracle VirtualBox

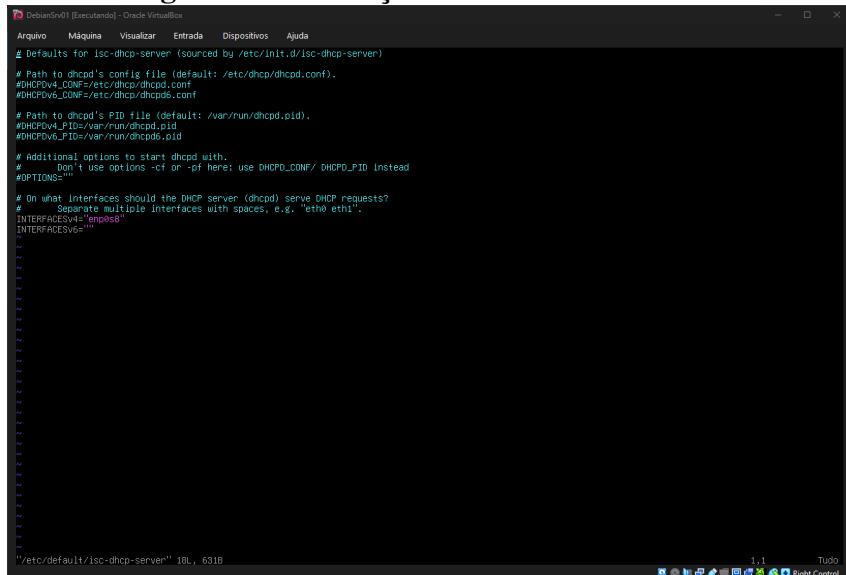
Arquivo Máquina Visualizar Entrada Dispositivos Ajuda

marmarins@debianrv:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback brd 00:00:00:00:00:00 state UNKNOWN
    inet 127.0.0.1/8 brd 00:00:00:00:00:00 scope host
        valid_lifetiime forever
        link_layer brd 00:00:00:00:00:00
        brd 00:00:00:00:00:00
        inet6 ::1/128 brd :: scope host
            valid_lifetiime forever
            link_layer brd ::00:00:00:00:00:00
            brd ::00:00:00:00:00:00
2: ensps3: <NO-CARRIER,BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:00:00:32 brd ff:ff:ff:ff:ff:ff
    altname enx000c29000032
    altname enx000c29000032
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enps3
        valid_lifetiime 86175sec preferred_lifetiime 73757sec
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic mngtmpaddr noprefixroute
        valid_lifetiime 60300sec preferred_lifetiime 14988sec
    inet6 fe80::100c:29ff:fe00:32/64 scope link
        valid_lifetiime forever
        link_layer brd ::00:00:00:00:00:00
        brd ::00:00:00:00:00:00
3: ensps1: <NO-CARRIER,BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:00:00:39 brd ff:ff:ff:ff:ff:ff
    altname enx000c29000039
    altname enx000c29000039
    inet 192.168.1.24 brd 192.168.1.255 scope global enps1
        valid_lifetiime forever
        preferred_lifetiime forever
    inet6 fe80::100c:29ff:fe00:39/64 scope link proto kernel ll
        valid_lifetiime forever
        preferred_lifetiime forever
marmarins@debianrv:~$
```

Fonte: Elaborado pelos autores

A Figura 33 demonstra a execução do comando `ip a` na VM DebianSrv, confirmando que as interfaces estão corretamente configuradas: `enp0s3` com IP dinâmico e `enp0s8` com IP fixo `192.168.1.1`.

Figura 34 – Definição da Interface do DHCP



The screenshot shows a terminal window titled "Debian9v01 [Executando] - Oracle VM VirtualBox". The window displays the contents of the file "/etc/default/isc-dhcp-server". The configuration includes paths for config files, PID files, and interface definitions. A specific line defines the interface "INTERFACESv4=enp0s8". The terminal window also shows the command "cat > /etc/default/isc-dhcp-server" followed by the file path and line numbers.

```
# Defaults for isc-dhcp-server (sourced by /etc/init.d/isc-dhcp-server)

# Path to dhcpd's config files (default: /etc/dhcp/dhcpd.conf).
#DHCPDv4_CONF=/etc/dhcp/dhcpd.conf
#DHCPDv6_CONF=/etc/dhcp/dhcpd6.conf

# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
#DHCPDv4_PID=/var/run/dhcpd.pid
#DHCPDv6_PID=/var/run/dhcpd6.pid

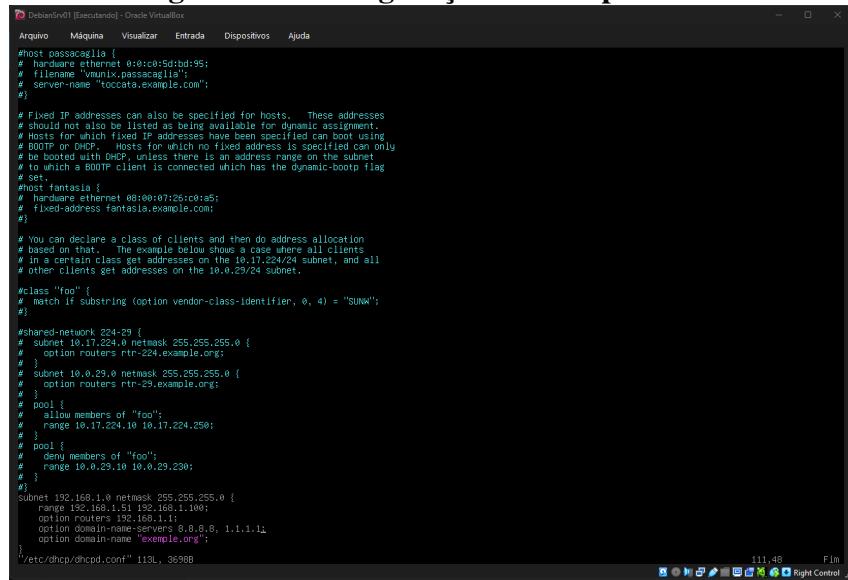
# Additional options to start dhcp with.
#       Don't use options -cf or -gf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="enp0s8"
INTERFACESv6=""

"/etc/default/isc-dhcp-server" 10L, 601B
```

Fonte: Elaborado pelos autores

A Figura 34 mostra o arquivo `/etc/default/isc-dhcp-server`, onde a interface de atendimento do DHCP foi definida como `enp0s8`, responsável pela distribuição de IPs na rede interna.

Figura 35 – Configuração do Escopo DHCP

```
#host passacaglia {
#    hardware ethernet 00:0c:05:bd:95;
#    filename 'vmunix.passacaglia';
#    server-name "toccata.example.com";
#}

# Fixed IP addresses can also be specified for hosts. These addresses
# should not also be listed as being available for dynamic assignment.
# Hosts for which fixed IP addresses have been specified can boot using
# BOOTP or DHCP. Hosts for which no fixed address is specified can only
# boot using DHCP. If no fixed address is specified there is an address range on the subnet
# to which a BOOTP client is connected which has the dynamic-bootp flag
# set.

#host fantasia {
#    hardware ethernet 00:0c:07:26:c9:5;
#    fixed-address fantasia.example.com;
#}

# You can declare a class of clients and then do address allocation
# based on that. The example below shows a case where all clients
# in a certain class get addresses on the 10.17.224/24 subnet, and all
# other clients get addresses on the 10.0.29/24 subnet.

#class "foo" {
#    match if substring(option vendor-class-identifier, 0, 4) = "SUNK";
#}

#shared-network 224-29 {
#    subnet 10.17.224.0 netmask 255.255.255.0 {
#        option routers rtt-224.example.org;
#    }
#    subnet 10.0.29.0 netmask 255.255.255.0 {
#        option routers rtt-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 192.168.1.0 netmask 255.255.255.0 {
#    range 192.168.1.100 192.168.1.100;
#    option routers 192.168.1.1;
#    option domain-name-servers 8.8.8.8, 1.1.1.1;
#    option domain-name "example.org";
#}

/etc/dhcp/dhcpd.conf" 113L, 3698B
```

Fonte: Elaborado pelos autores

A Figura 35 exibe o conteúdo do arquivo `/etc/dhcp/dhcpd.conf`, com a sub-rede 192.168.1.0/24, faixa de IPs de 192.168.1.51 a 192.168.1.100, gateway 192.168.1.1 e servidores DNS 8.8.8.8 e 1.1.1.1.

Figura 36 – Reinicialização e Status do Serviço DHCP

The screenshot shows a terminal window titled "Debian GNU [Executando] - Oracle VirtualBox". The window contains the following text:

```
Arquivo Máquina Visualizar Entrada Dispositivos Ajuda
onmartins@debiantsrv:~$ sudo systemctl restart isc-dhcp-server
onmartins@debiantsrv:~$ sudo systemctl status isc-dhcp-server
● isc-dhcp-server.service - isc-dhcp-server; generated
  Loaded: loaded (/etc/init.d/isc-dhcp-server; generated)
    Active: active (running) since Sun 2025-10-19 01:11:56 -03; 10s ago
      Invocation: f82c3e0b5f1fe480000c5110650b6d000
    Process: 1596 ExecStart=/etc/init.d/isc-dhcp-server start (code=exited, status=0/SUCCESS)
   Tasks: 1 (limit: 2040)
  Memory: 1.9M (peak: 9.3M)
        CPU: 5ms
     CGroup: /system.slice/isc-dhcp-server.service
             └─1086 /usr/sbin/dhcpd -4 -q -cf /etc/dhcp/dhcpd.conf enp0s8

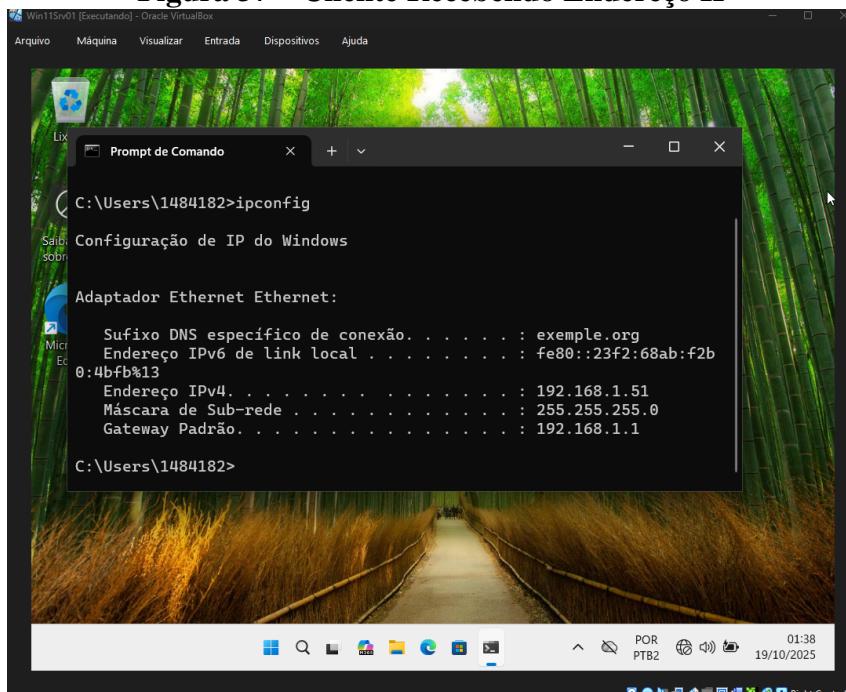
out 19 01:11:54 debiantsrv systemd[1]: Starting isc-dhcp-server.service - LSB: DHCP server...
out 19 01:11:54 debiantsrv isc-dhcp-server[1596]: Launching IPv4 server only.
out 19 01:11:54 debiantsrv dhcpcd[1582]: Wrote 1 leases to leases file.
out 19 01:11:54 debiantsrv dhcpcd[1582]: Starting DHCPv4 server...
out 19 01:11:54 debiantsrv isc-dhcp-server[1596]: Started ISC DHCPv4 server: dhcpd.
out 19 01:11:56 debiantsrv systemd[1]: Started isc-dhcp-server.service - LSB: DHCP server.

onmartins@debiantsrv:~$ "
```

Fonte: Elaborado pelos autores

A Figura 36 demonstra os comandos de reinicialização e verificação do serviço `isc-dhcp-server`, confirmando que o servidor DHCP está ativo e funcional.

Figura 37 – Cliente Recebendo Endereço IP



Fonte: Elaborado pelos autores

A Figura 37 apresenta a VM Windows 11 conectada à rede interna, executando o comando `ipconfig`. O cliente recebeu automaticamente um IP dentro da faixa definida, validando o funcionamento do servidor DHCP.

3.25 Servidor Web e banco de dados

O servidor web foi implementado na AWS com o protocolo HTTPS, garantindo comunicações seguras entre clientes e sistemas internos. Ele hospeda aplicações corporativas e o site institucional da LOGAM Tech. As principais configurações podem ser vistas a seguir:

Tabela 10 – Informações do servidor (AWS)

Nome do Servidor	Endereço IP Público	Endereço IP Privado	Usuário de Acesso	Função
Servidor Web + Banco de Dados	54.145.137.231	172.31.16.36	N/A	Configuração de servidor web com banco de dados para realizar um CRUD simples.

Fonte: Criado pelos autores

3.26 Configurar Security Groups

3.26.1 *Security Group para Servidor Web*

- Nome: Web
- Descrição: Security Group para servidor web
- Regras de entrada:
 - SSH: 0.0.0.0/0:22
 - HTTP: 0.0.0.0/0:80
 - HTTPS: 0.0.0.0/0:443
- Regras de saída:
 - Todos: 0.0.0.0/0:0

3.26.2 *Security Group para Banco de Dados*

- Nome: Database

- Descrição: Security Group para banco de dados
- Regras de entrada:
 - PostgreSQL do security group Web: 0.0.0.0/0:5432 (Observação: O ideal é restringir a entrada apenas ao Security Group Web ao invés de 0.0.0.0/0)
- Regras de saída:
 - Todos: 0.0.0.0/0

3.27 Criar o RDS PostgreSQL

- Nome: database-0
- Engine: PostgreSQL
- Versão: 17.4-R1
- Template: Sandbox
- Instance class: db.t4g.micro
- Storage type: gp2
- Storage: 20GB
- Security Group: Database
- Initial database: test_database

3.28 Criar o EC2 para o Servidor Web

- Nome: web-server
- Sistema operacional: Ubuntu 24.04 LTS
- Tipo de instância: t3.micro
- Security Group: Web
- Storage: 8GB

3.28.1 *Configurar Elastic IP para o EC2*

Objetivo: Garantir que o servidor web tenha um endereço IP público estático, evitando mudanças de IP quando a instância for reiniciada.

a) Alocar Elastic IP:

- Acessar o console AWS EC2
- Navegar para “Elastic IPs” no menu lateral
- Clicar em “Allocate Elastic IP address”

b) Associar Elastic IP à instância EC2 **servidor-web**:

- Selecionar o Elastic IP recém-criado
- Selecionar a instância **servidor-web**

c) Verificar associação:

- Confirmar que o Elastic IP está associado à instância
- Anotar o endereço IP público estático para uso posterior

3.29 Configurar o Servidor Web

3.29.1 *Acessar a instância EC2 **servidor-web***

```
1 ssh -i "[chave-ssh]" ubuntu@54.145.137.231
```

3.29.2 *Clonar esse repositório e executar o script de instalação do Docker*

```
1 mkdir app
2 cd app
3 git clone [url-do-repositorio] .
4 cd servidor-web-e-banco-de-dados
5 sudo ./install-docker-ubuntu.sh
```

Sair da sessão SSH e entrar novamente para aplicar as alterações.

3.29.3 Configurar o ambiente

```
1 cp env.production.example .env
```

3.29.4 Editar o arquivo .env com as credenciais do RDS

```
1 vim .env
2 # Editar o arquivo .env com as credenciais do RDS
```

3.29.5 Deploy

```
1 docker-compose -f docker-compose.prod.yml --env-file .env up -d --build
```

3.29.6 Atualizar security group do servidor web

Adicionar regra de entrada para porta 3000 do security group do servidor web para que seja possível acessar as rotas da API pela internet usando o Elastic IP.

3.30 Testar a API

Observação: Os comandos `curl` abaixo devem ser executados a partir de um terminal externo (fora do EC2), utilizando o Elastic IP.

3.30.1 Teste de saúde

```
1 curl --location '54.145.137.231:3000/health' \
2 --header 'Content-Type: application/json' \
3 --data ''
```

3.30.2 Teste de usuários

Criar Usuário:

```
1 curl --location '54.145.137.231:3000/api/users' \
2 --header 'Content-Type: application/json' \
3 --data-raw '{"name": "John Doe", "email": "john@example.com", "age": 30}'
```

Listar Usuários:

```
1 curl --location '54.145.137.231:3000/api/users' \
2 --header 'Content-Type: application/json'
```

3.30.3 Teste de produtos

Criar Produto:

```
1 curl --location '54.145.137.231:3000/api/products' \
2 --header 'Content-Type: application/json' \
3 --data-raw '{"name": "Product 1", "description": "Description 1", "price": \
100, "stock": 10}'
```

Listar Produtos:

```
1 curl --location '54.145.137.231:3000/api/products' \
2 --header 'Content-Type: application/json'
```

3.31 Configurar o Nginx

```
1 sudo apt update
2 sudo apt install -y nginx
3 sudo systemctl start nginx
4 sudo systemctl enable nginx
5 sudo apt install make -y
6 make nginx-setup
```

Detalhes do que o comando `make nginx-setup` configura:

O comando `make nginx-setup` executa o script `update-nginx-config.sh` que realiza as seguintes configurações:

- a) **Verificação de segurança:** Verifica se o script está sendo executado com privilégios de root/sudo e valida a existência dos arquivos necessários (`nginx.config` e `index.html`).
- b) **Backup da configuração atual:** Cria diretório de backup (`/etc/nginx/backups`) e faz backup da configuração atual com timestamp.

- c) **Aplicação da nova configuração:** Substitui /etc/nginx/sites-available/default com o conteúdo de nginx.config e testa a configuração com nginx -t.
- d) **Configuração do Nginx (nginx.config):**
- Servidor HTTP na porta 80.
 - Diretório raiz: /var/www/app.
 - Servir arquivos estáticos: Configuração para servir arquivos HTML/CSS/JS.
 - Proxy para API: Redireciona requisições /api/ para http://127.0.0.1:3000/api/.
 - Health check: Redireciona /health para http://127.0.0.1:3000/health.
 - Headers de proxy: Configura headers necessários para proxy reverso.
- e) **Criação do diretório web:** Cria diretório /var/www/app, define proprietário (www-data:www-data) e permissões (755).
- f) **Cópia dos arquivos estáticos:** Copia index.html para /var/www/app/ e define permissões.
- g) **Recarga do Nginx:** Executa systemctl reload nginx para aplicar as mudanças e verifica o sucesso da recarga.

Estrutura final da configuração:

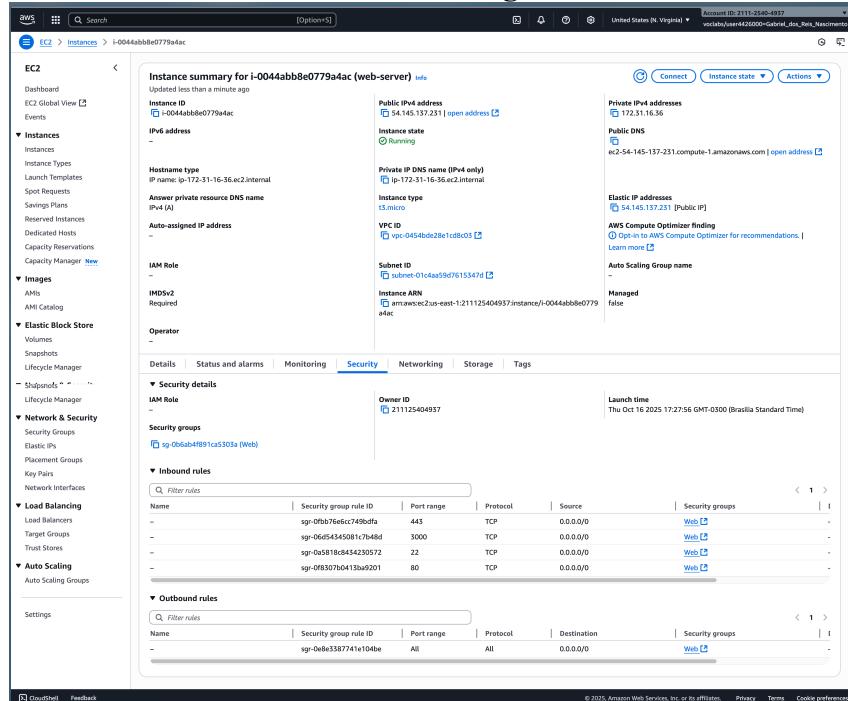
- **Frontend:** Servido diretamente pelo Nginx na porta 80.
- **API:** Proxy reverso para aplicação Node.js na porta 3000.
- **Arquivos estáticos:** Servidos do diretório /var/www/app.

3.32 Testar no navegador

- a) Acessar o servidor web via browser no endereço http://54.145.137.231
- b) Validar que a interface de CRUD está sendo exibida corretamente.
- c) Testar adicionar usuários e produtos.

As figuras a seguir demonstram o funcionamento do ambiente de servidor web e banco de dados configurado na AWS. Cada etapa evidencia um componente da arquitetura e comprova o correto funcionamento dos serviços, desde a infraestrutura e segurança até os testes de integração entre API e banco de dados.

Figura 38 – Painel da Instância EC2 na AWS com o Elastic IP associado e Security Group configurado



Fonte: Elaborado pelos autores

A Figura 338 apresenta a instância EC2 configurada para hospedar o servidor web. Essa instância executa o sistema operacional Ubuntu 24.04 LTS e possui um Elastic IP, garantindo endereço público fixo mesmo após reinicializações. É nela que residem o Nginx e a aplicação Node.js, responsáveis por servir o frontend e a API do sistema CRUD.

Figura 39 – Painel do Banco de Dados RDS PostgreSQL com o endpoint e Security Group configurado

Fonte: Elaborado pelos autores

A Figura 39 exibe o serviço Amazon RDS utilizado para o banco de dados da aplicação. O banco de dados PostgreSQL foi criado com acesso restrito apenas à instância EC2, por meio de um Security Group dedicado, assegurando que as conexões externas sejam bloqueadas. Esse banco armazena as tabelas utilizadas pelo sistema CRUD (como usuários e produtos).

Figura 40 – Regras de Segurança do Servidor Web (Security Group)

Inbound rules (4)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
sgr-0fb76fc749bdfa	sgr-0fb76fc749bdfa	IPv4	HTTPS	TCP	443	0.0.0.0/0	-
sgr-06f54545081c7b48d	sgr-06f54545081c7b48d	IPv4	Custom TCP	TCP	3000	0.0.0.0/0	-
sgr-05f818c8434230572	sgr-05f818c8434230572	IPv4	SSH	TCP	22	0.0.0.0/0	-
sgr-0f8307b0413ba9201	sgr-0f8307b0413ba9201	IPv4	HTTP	TCP	80	0.0.0.0/0	-

Outbound rules (1)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Destination	Description
sgr-0e8c3387741e104be	sgr-0e8c3387741e104be	IPv4	All traffic	All	All	0.0.0.0/0	-

Fonte: Elaborado pelos autores

A Figura 40 mostra as regras de segurança aplicadas ao servidor web. Apenas as portas necessárias foram liberadas: SSH (22) para administração, HTTP (80) e HTTPS (443) para acesso público, e 3000 para comunicação com a API Node.js. Para saída, todas as portas (0.0.0.0/0) estão liberadas, permitindo que o servidor faça requisições externas conforme necessário.

Figura 41 – Regras de Segurança do Banco de Dados (Security Group)

The figure consists of two vertically stacked screenshots of the AWS VPC Security Groups interface. Both screenshots show a security group named 'sg-0991009d097c7c009 - Database'.

Screenshot 1: Inbound Rules (2)

Name	Security group rule ID	Type	Protocol	Port range	Source	Description
sgr-0215b1261f506d403	-	PostgreSQL	TCP	5432	so-065ab4f891ca5303...	EC2
sgr-0a5cad54b861b831d	IPv4	PostgreSQL	TCP	5432	192.141.114.204/32	Gab

Screenshot 2: Outbound Rules (1)

Name	Security group rule ID	Type	Protocol	Port range	Destination	Description
sgr-08f1af8878b2db8b	IPv4	All traffic	All	All	0.0.0.0/0	-

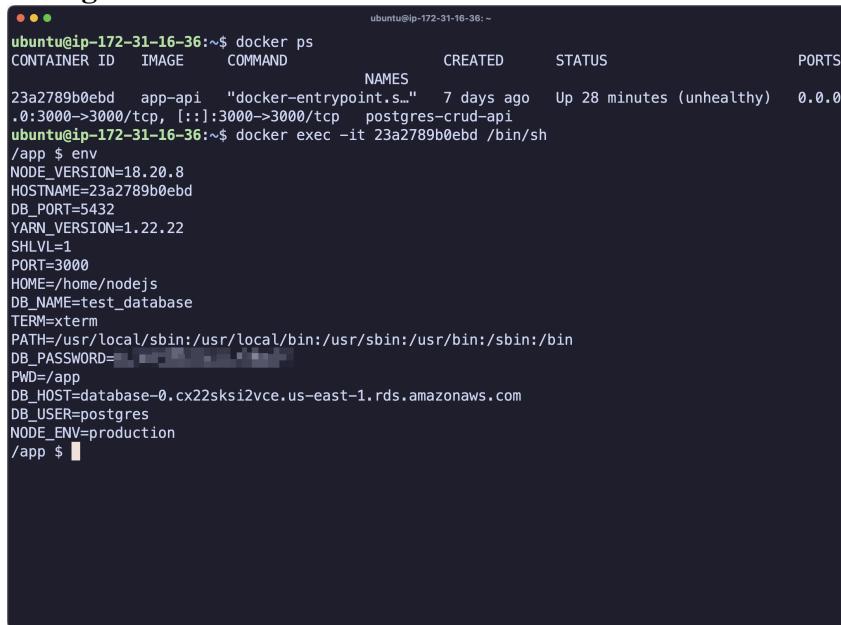
Fonte: Elaborado pelos autores

A Figura 41 apresenta o Security Group do banco de dados PostgreSQL. Ele permite conexões apenas provenientes do grupo de segurança do servidor web, na porta 5432, bloqueando todo o tráfego externo. Para saída, todas as portas (0.0.0.0/0) estão liberadas, permitindo que o servidor faça requisições externas conforme necessário.

Figura 42 – Acesso SSH ao Servidor Web

Fonte: Elaborado pelos autores

A Figura 42 demonstra o acesso SSH ao servidor EC2 e a execução do comando docker ps e o comando docker logs [nome-do-container] para verificar os logs do container da aplicação Node.js em execução. Evidenciando que o container da aplicação Node.js está em execução e está funcionando corretamente.

Figura 43 – Variáveis de Ambiente no Container da API

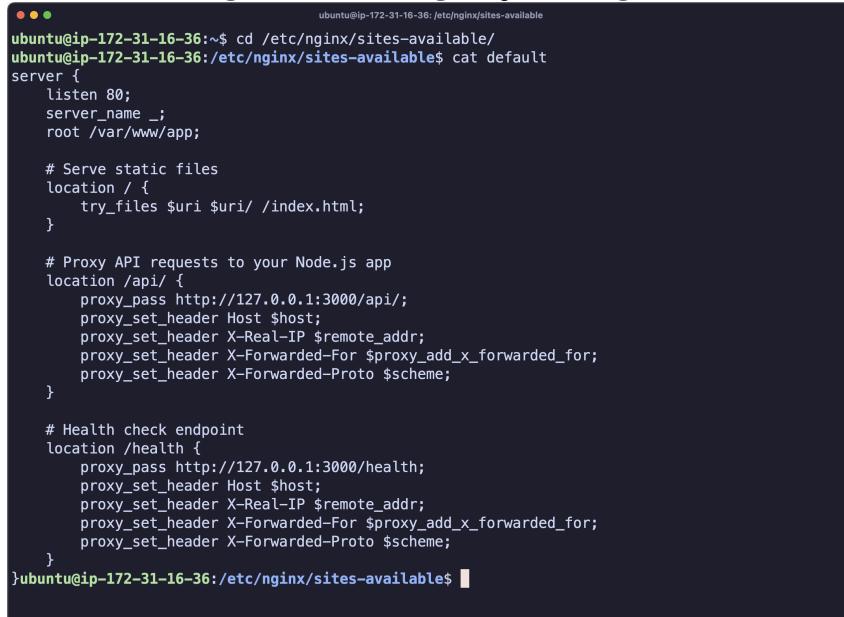
A screenshot of a terminal window titled "ubuntu@ip-172-31-16-36:~". The window displays the output of several commands: "docker ps" showing a single container named "app-api" with port mappings; "docker exec -it 23a2789b0ebd /bin/sh" entering the container; "env" command listing environment variables; and "cat .env" command showing a file with environment variable definitions.

```
ubuntu@ip-172-31-16-36:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS          PORTS
23a2789b0ebd   app-api   "docker-entrypoint.s..."  7 days ago    Up 28 minutes (unhealthy)   0.0.0
.0:3000->3000/tcp, [::]:3000->3000/tcp   postgres-crud-api

ubuntu@ip-172-31-16-36:~$ docker exec -it 23a2789b0ebd /bin/sh
/app $ env
NODE_VERSION=18.20.8
HOSTNAME=23a2789b0ebd
DB_PORT=5432
YARN_VERSION=1.22.22
SHLVL=1
PORT=3000
HOME=/home/nodejs
DB_NAME=test_database
TERM=xterm
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
DB_PASSWORD=[REDACTED]
PWD=/app
DB_HOST=database-0.cx22sksi2vce.us-east-1.rds.amazonaws.com
DB_USER=postgres
NODE_ENV=production
/app $
```

Fonte: Elaborado pelos autores

A Figura 43 mostra o acesso ao container da API Node.js via docker exec e a execução do comando env, exibindo as variáveis de ambiente configuradas dentro do container em execução. Essas variáveis incluem as credenciais e configurações necessárias para conexão com o banco de dados PostgreSQL no RDS.

Figura 44 – Configuração do Nginx

```
ubuntu@ip-172-31-16-36:~$ cd /etc/nginx/sites-available/
ubuntu@ip-172-31-16-36:/etc/nginx/sites-available$ cat default
server {
    listen 80;
    server_name _;
    root /var/www/app;

    # Serve static files
    location / {
        try_files $uri $uri/ /index.html;
    }

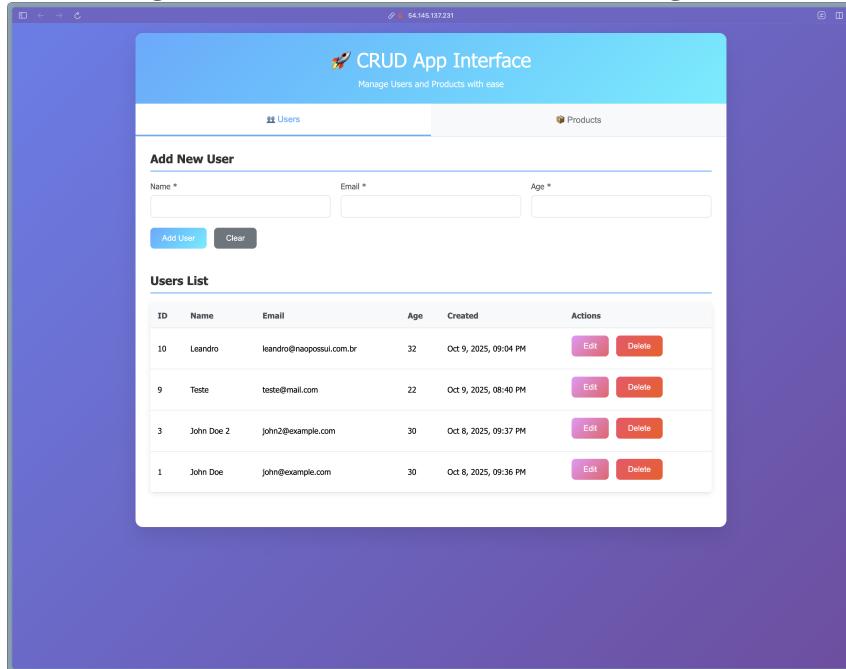
    # Proxy API requests to your Node.js app
    location /api/ {
        proxy_pass http://127.0.0.1:3000/api/;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    # Health check endpoint
    location /health {
        proxy_pass http://127.0.0.1:3000/health;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}ubuntu@ip-172-31-16-36:/etc/nginx/sites-available$
```

Fonte: Elaborado pelos autores

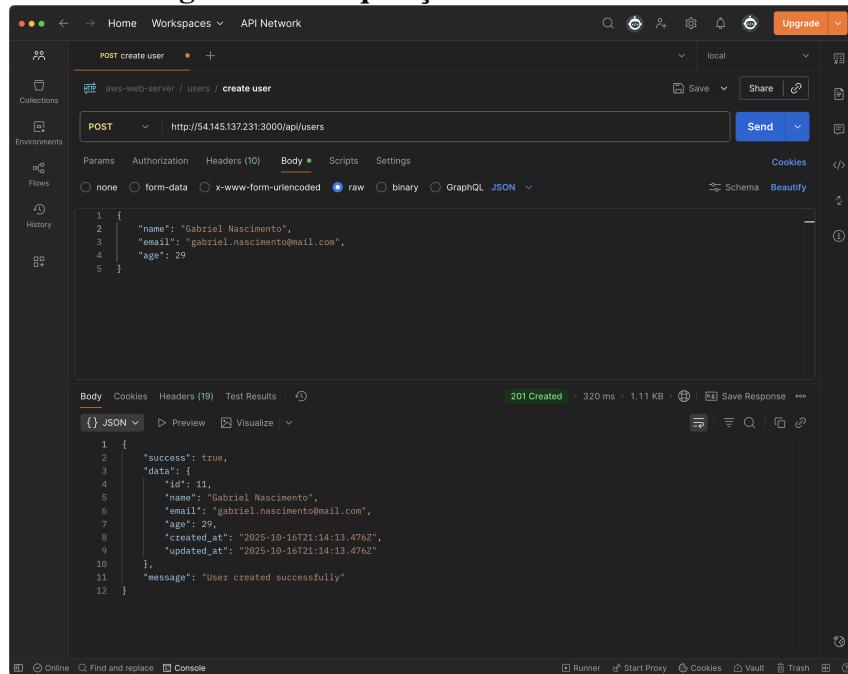
A Figura 44 mostra a configuração do Nginx utilizada como proxy reverso. Essa configuração direciona as requisições HTTP da rota /api para a aplicação Node.js interna, enquanto serve os arquivos estáticos do frontend diretamente.

Figura 45 – Interface do CRUD no Navegador



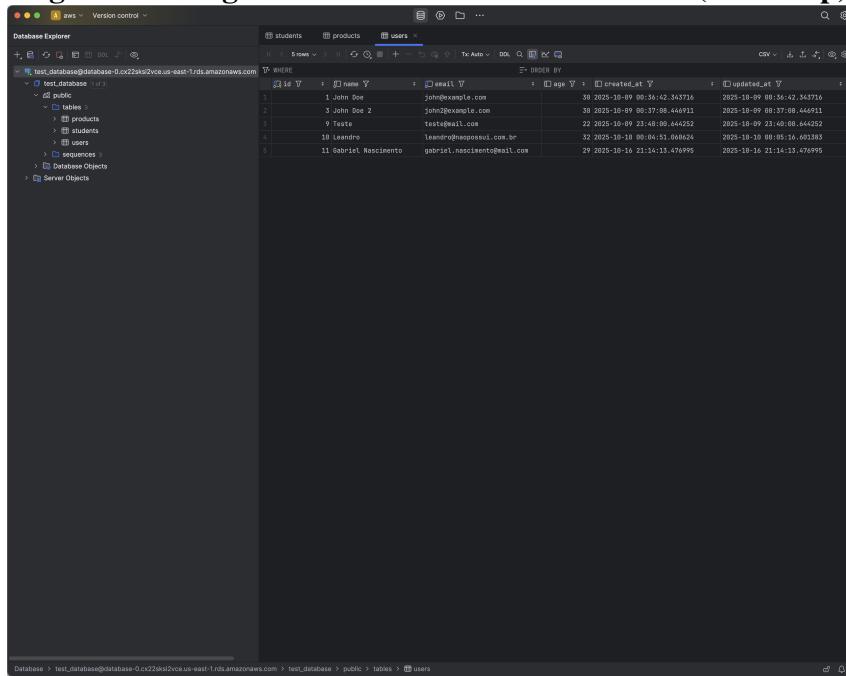
Fonte: Elaborado pelos autores

A Figura 45 apresenta a interface do sistema CRUD sendo servida pelo Nginx, acessada diretamente através do Elastic IP da instância EC2. Essa tela comprova que o servidor web está ativo e a aplicação está disponível publicamente.

Figura 46 – Requisição POST no Postman

Fonte: Elaborado pelos autores

A Figura 46 demonstra o envio de uma requisição POST para a API utilizando o Postman. O retorno JSON confirma que o servidor web recebeu a requisição, processou os dados e gravou o novo registro no banco de dados PostgreSQL.

Figura 47 – Registro Inserido no Banco de Dados (DataGrip)

The screenshot shows the DataGrip interface with the 'Database Explorer' tab selected. Under the 'test_database' node, the 'public' schema is expanded, revealing the 'users' table. The 'users' table has columns: id, name, email, age, created_at, and updated_at. There are 11 rows of data:

		name	email	age	created_at	updated_at
1	1	John Doe	john@example.com	30	2025-10-09 00:36:42.5437316	2025-10-09 08:36:42.543716
2	2	John Doe 2	john2@example.com	30	2025-10-09 00:37:08.446911	2025-10-09 08:37:08.446911
3	3	Teste	teste@gmail.com	22	2025-10-09 23:40:00.444252	2025-10-09 23:40:00.444252
4	4	Leandro	leandro@nascimentou.com.br	32	2025-10-10 00:04:51.048424	2025-10-10 00:05:16.403383
5	5	Gabriel Nascimento	gabriel.nascimento@email.com	29	2025-10-16 21:14:13.476995	2025-10-16 21:14:13.476995

Fonte: Elaborado pelos autores

A Figura 47 exibe o banco de dados acessado via DataGrip, comprovando que o registro inserido através da requisição POST na API na Figura 9 foi armazenado na tabela users. Essa evidência confirma a integração entre o servidor web na EC2 e o banco de dados PostgreSQL no RDS.