Configurações Organizacionais

```
# Configurações para o Grupo AD2. Substitua Pedro
# pelo seu domínio e ajuste os DNs conforme necessário.
config set force_ldap_ssl true
config set force_ldap_port 636
```

- Define que o Cockpit deve usar apenas conexões LDAPS (SSL) para autenticar usuários.
- Força o Cockpit a se conectar à porta 636 do servidor LDAP, garantindo que a senha do usuário não trafegue em texto puro na rede.

Carrinho de compras - Integração(Prestamos de Produtos/AD)

```
# Configurações para integrar Cockpit com Grupo AD2
# Substitua Pedro pelo seu domínio.
cockpit -d
```

- Inicia o Cockpit em modo de depuração (-d), exibindo mais detalhes no terminal para facilitar a identificação de erros de conexão e autenticação.

Desativar painel cockpit/Ativar painel

```
configure-cockpit-panel --"<regex>" disable/ enable
```

- Permite ativar ou desativar o painel do Cockpit para usuários específicos, com base em uma expressão regular (regex) fornecida.
- Útil para controlar quais usuários podem acessar o painel de administração do Cockpit.

Configuração de Armazenamento de Volumes/Pools

```
config set pool-ssd /dev/sdb1
config set pool-hdd /dev/sdc1
# Criar pools gerenciados
cockpit-storaged-pool create ssd /dev/sdb1
cockpit-storaged-pool create hdd /dev/sdc1
```

- Define os dispositivos físicos (/dev/sdb1 e /dev/sdc1) que serão usados para criar os pools de armazenamento SSD e HDD.
- Cria os pools de armazenamento gerenciados pelo Cockpit, permitindo a criação de volumes e sistemas de arquivos sobre esses discos.

Ativação dos Bitwarden Firmas

```
# Criar diretórios para os dados persistentes
mkdir -p /opt/bitwarden/{mssql,data,logs}
# Ajustar permissões (o container roda como root)
chown -R root:root /opt/bitwarden
```

- Criação dos diretórios para armazenar os dados do SQL Server (mssql), dados da aplicação (data) e logs (logs).
- Ajuste de permissões para que o container possa escrever nos diretórios montados.

SS-[illegible]

```
# Gerar senha forte para o SA
openssl rand -base64 32 > /opt/bitwarden/mssql/sa.pass
# Criar arquivo de variáveis de ambiente
cat > /opt/bitwarden/.env <<EOF
MSSQL_SA_PASSWORD=$(cat /opt/bitwarden/mssql/sa.pass)
ACCEPT_EULA=Y
MSSQL_PID=Express
EOF
```

- Gera uma senha forte para o usuário SA do SQL Server e salva em /opt/bitwarden/mssql/sa.pass.
- Cria o arquivo .env com as variáveis de ambiente necessárias para o container do SQL Server, incluindo a senha gerada, aceitação da EULA e versão Express.

Rhumsaa/azuredatastudio — Instalação e configuração no Ubuntu/Linux

```
# Instalar dependências
sudo apt-get update
sudo apt-get install -y libgtk-3-0 libnotify4 libnss3 libxss1 libxtst6 libasound2
# Baixar o .deb mais recente (ajuste a versão se quiser fixar)
wget -O /tmp/azuredatastudio.deb \
  "https://go.microsoft.com/fwlink/?linkid=2177931"
# Instalar
sudo dpkg -i /tmp/azuredatastudio.deb
sudo apt-get -f install
# Criar atalho no menu
cat > ~/.local/share/applications/azuredatastudio.desktop <<EOF
[Desktop Entry]
Name=Azure Data Studio
Comment=Azure Data Studio
Exec=/usr/share/azuredatastudio/azuredatastudio --no-sandbox %U
Icon=azuredatastudio
Terminal=false
Type=Application
Categories=Development;Database;
EOF
chmod +x ~/.local/share/applications/azuredatastudio.desktop
# Atualizar cache do menu
update-desktop-database ~/.local/share/applications
```

- Instala as dependências gráficas e de rede necessárias para o Azure Data Studio rodar no Ubuntu/Linux.
- Baixa o pacote .deb mais recente do Azure Data Studio diretamente do site da Microsoft.
- Instala o pacote e corrige dependências faltantes, se necessário.
- Cria um atalho no menu de aplicativos do usuário, permitindo abrir o Azure Data Studio sem precisar usar o terminal.
- Atualiza o cache do menu de aplicativos para o atalho aparecer imediatamente.

Arquivo

Configs

```
# Gerar configuração base para editar
/usr/share/cockpit/scripts/gen-cockpit-conf > /tmp/cockpit.conf
# Editar
nano /tmp/cockpit.conf
# Instalar no local oficial
sudo install -m 644 -o root -g root /tmp/cockpit.conf /etc/cockpit/cockpit.conf
# Reiniciar o Cockpit para aplicar
sudo systemctl restart cockpit.socket
# Verificar status
sudo systemctl status cockpit.socket --no-pager
```

- Gera um arquivo de configuração base do Cockpit em /tmp/cockpit.conf, pronto para ser editado antes de ser instalado no local oficial.
- Instala o arquivo de configuração em /etc/cockpit/cockpit.conf com permissões root:root e modo 644, garantindo que apenas o root possa editar.
- Reinicia o socket do Cockpit para que as novas configurações sejam carregadas.
- Exibe o status do socket do Cockpit, confirmando se ele está ativo e rodando corretamente.

```
# Criar diretório do Chrony
mkdir -p /opt/chrony/{data,logs}
# Ajustar permissões
chown -R root:root /opt/chrony
# Criar arquivo de configuração
cat > /opt/chrony/chrony.conf <<EOF
# Servidores NTP brasileiros
pool a.ntp.br iburst
pool b.ntp.br iburst
pool c.ntp.br iburst
# Permitir clientes da rede local
allow 10.0.0.0/8
allow 192.168.0.0/16
allow 172.16.0.0/12
# Arquivos de log
logdir /var/log/chrony
# Arquivo de estatísticas
dumpdir /var/lib/chrony
# Sincronizar relógio do hardware
rtcsync
# Permitir ajuste rápido no início
makestep 1.0 3
EOF
```

- Cria o diretório /opt/chrony/ com subpastas para dados (data) e logs (logs).
- Ajusta as permissões do diretório para root:root, garantindo que apenas o root possa editar os arquivos.
- Gera o arquivo de configuração do Chrony (/opt/chrony/chrony.conf), definindo os servidores NTP brasileiros (a.ntp.br, b.ntp.br, c.ntp.br), as redes que podem se sincronizar (10.0.0.0/8, 192.168.0.0/16, 172.16.0.0/12), os caminhos de log e estatísticas, e ativar sincronização do relógio do hardware e ajuste rápido no início.

Configurar servidor de horas do Brasil

```
# Criar diretório
mkdir -p /opt/chrony/{data,logs}
# Ajustar permissões
chown -R root:root /opt/chrony
# Criar arquivo de configuração
cat > /opt/chrony/chrony.conf <<EOF
# Servidores NTP brasileiros
pool a.ntp.br iburst
pool b.ntp.br iburst
pool c.ntp.br iburst
# Permitir clientes da rede local
allow 10.0.0.0/8
allow 192.168.0.0/16
allow 172.16.0.0/12
# Arquivos de log
logdir /var/log/chrony
# Arquivo de estatísticas
dumpdir /var/lib/chrony
# Sincronizar relógio do hardware
rtcsync
# Permitir ajuste rápido no início
makestep 1.0 3
EOF
```

- Cria o diretório /opt/chrony/ com subpastas para dados (data) e logs (logs).
- Ajusta as permissões do diretório para root:root, garantindo que apenas o root possa editar os arquivos.
- Gera o arquivo de configuração do Chrony (/opt/chrony/chrony.conf), definindo os servidores NTP brasileiros (a.ntp.br, b.ntp.br, c.ntp.br), as redes que podem se sincronizar (10.0.0.0/8, 192.168.0.0/16, 172.16.0.0/12), os caminhos de log e estatísticas, e ativar sincronização do relógio do hardware e ajuste rápido no início.

Sincronizar horário c/ Chrony

```
# Instalar
sudo apt-get install -y chrony
# Ativar e iniciar
sudo systemctl enable --now chrony
# Forçar sincronização imediata
sudo chronyc makestep
# Verificar fontes
chronyc sources -v
# Verificar tracking
chronyc tracking
```

- Instala o Chrony, que substitui o ntpdate de forma mais moderna e eficiente.
- Habilita e inicia o serviço chrony imediatamente, e também o coloca para iniciar automaticamente a cada boot.
- Força uma sincronização imediata do relógio do sistema com os servidores NTP configurados.
- Mostra os servidores NTP ativos, com detalhes (-v) sobre qualidade, latência e estratos.
- Exibe o status da sincronização do relógio, mostrando o servidor atual, o desvio do relógio e a precisão.

Criar grupo Sphi

```
# Criar o grupo
sudo groupadd sphi
# Adicionar usuários ao grupo
sudo usermod -aG sphi usuario1
sudo usermod -aG sphi usuario2
# Verificar membros do grupo
getent group sphi
```

- Cria um grupo de sistema chamado Sphi, caso não exista.
- Adiciona os usuários usuario1 e usuario2 ao grupo Sphi, mantendo os grupos já existentes (-aG).
- Exibe os membros do grupo Sphi, confirmando que os usuários foram adicionados corretamente.

Criar usuários Sudo/não sudo

```
# Criar usuário com sudo
sudo useradd -m -s /bin/bash -G sudo usuario1
# Criar usuário sem sudo
sudo useradd -m -s /bin/bash usuario2
# Definir senhas
sudo passwd usuario1
sudo passwd usuario2
# Verificar grupos
groups usuario1
groups usuario2
# Remover sudo de um usuário
sudo deluser usuario1 sudo
```

- Cria dois usuários: usuario1 com sudo e usuario2 sem sudo.
- Adiciona usuario1 ao grupo sudo, concedendo permissões de administrador.
- Define senhas para os usuários.
- Exibe os grupos de cada usuário, confirmando quem tem sudo e quem não tem.
- Remove usuario1 do grupo sudo, revogando suas permissões de administrador.

Gerar chave SSH para Sudo/não sudo

```
# Gerar chave para usuário com sudo
sudo -u usuario1 ssh-keygen -t ed25519 -C "usuario1@$(hostname)" -f /home/usuario1/.ssh/id_ed25519 -N ""
# Gerar chave para usuário sem sudo
sudo -u usuario2 ssh-keygen -t ed25519 -C "usuario2@$(hostname)" -f /home/usuario2/.ssh/id_ed25519 -N ""
# Ajustar permissões
chmod 700 /home/usuario1/.ssh
chmod 700 /home/usuario2/.ssh
chmod 600 /home/usuario1/.ssh/id_ed25519
chmod 600 /home/usuario2/.ssh/id_ed25519
# Exibir chaves públicas
cat /home/usuario1/.ssh/id_ed25519.pub
cat /home/usuario2/.ssh/id_ed25519.pub
```

- Gera uma chave SSH para usuario1 (com sudo) e usuario2 (sem sudo), usando o algoritmo ed25519 e o comentário identificando o usuário e o hostname.
- Ajusta as permissões dos diretórios .ssh para 700 e das chaves privadas para 600, garantindo que apenas o dono possa ler e escrever.
- Exibe as chaves públicas geradas, para que possam ser copiadas e adicionadas em servidores remotos.

Copiar chave pública para authorized_keys

```
# Copiar chave pública para authorized_keys
cat /home/usuario1/.ssh/id_ed25519.pub >> /home/usuario1/.ssh/authorized_keys
cat /home/usuario2/.ssh/id_ed25519.pub >> /home/usuario2/.ssh/authorized_keys
# Ajustar permissões
chmod 600 /home/usuario1/.ssh/authorized_keys
chmod 600 /home/usuario2/.ssh/authorized_keys
# Verificar conteúdo
cat /home/usuario1/.ssh/authorized_keys
cat /home/usuario2/.ssh/authorized_keys
```

- Adiciona a chave pública de usuario1 ao arquivo authorized_keys, permitindo que ele faça login sem senha.
- Adiciona a chave pública de usuario2 ao arquivo authorized_keys, permitindo que ele faça login sem senha.
- Ajusta as permissões dos arquivos authorized_keys para 600, garantindo que apenas o dono possa ler e escrever.
- Exibe o conteúdo dos arquivos authorized_keys, para confirmar que as chaves foram adicionadas corretamente.

Configurar Sponsor do acho no SSH

```
# Criar diretório
mkdir -p /root/.ssh
# Adicionar chave pública
echo "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIJ7ZqXmPqRsTuVwXyZaBcDeFgHiJkLmNoPqRsTuVwXy sponsor@acho" >> /root/.ssh/authorized_keys
# Ajustar permissões
chmod 700 /root/.ssh
chmod 600 /root/.ssh/authorized_keys
# Verificar
cat /root/.ssh/authorized_keys
```

- Cria o diretório /root/.ssh, caso não exista, com permissão 700 (apenas root pode acessar).
- Adiciona a chave pública ed25519 do sponsor@acho ao arquivo authorized_keys, permitindo que o usuário sponsor faça login como root sem senha.
- Ajusta as permissões do arquivo authorized_keys para 600, garantindo que apenas o root possa ler e escrever.
- Exibe o conteúdo do arquivo authorized_keys, para confirmar que a chave foi adicionada corretamente.

Apontar SSH para user Sponsor

```
# Criar ou editar /root/.ssh/config
cat >> /root/.ssh/config <<EOF
Host *
    User sponsor
EOF
# Ajustar permissões
chmod 600 /root/.ssh/config
# Verificar
cat /root/.ssh/config
```

- Adiciona ao arquivo /root/.ssh/config a regra para que todas as conexões SSH (Host *) usem o usuário sponsor por padrão.
- Ajusta as permissões do arquivo para 600, garantindo que apenas o root possa ler e escrever.
- Exibe o conteúdo do arquivo, para confirmar que a configuração foi adicionada corretamente.

Configurações de segurança do SSH

```
# Criar diretório
mkdir -p /root/.ssh
# Adicionar chave pública
echo "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIJ7ZqXmPqRsTuVwXyZaBcDeFgHiJkLmNoPqRsTuVwXy sponsor@acho" >> /root/.ssh/authorized_keys
# Ajustar permissões
chmod 700 /root/.ssh
chmod 600 /root/.ssh/authorized_keys
# Verificar
cat /root/.ssh/authorized_keys
```

- Cria o diretório /root/.ssh, caso não exista, com permissão 700 (apenas root pode acessar).
- Adiciona a chave pública ed25519 do sponsor@acho ao arquivo authorized_keys, permitindo que o usuário sponsor faça login como root sem senha.
- Ajusta as permissões do arquivo authorized_keys para 600, garantindo que apenas o root possa ler e escrever.
- Exibe o conteúdo do arquivo authorized_keys, para confirmar que a chave foi adicionada corretamente.

Instalar chrony NTP

```
# Instalar
sudo apt-get install -y chrony
# Ativar e iniciar
sudo systemctl enable --now chrony
# Forçar sincronização imediata
sudo chronyc makestep
# Verificar fontes
chronyc sources -v
# Verificar tracking
chronyc tracking
```

- Instala o chrony, que substitui o ntpdate de forma mais moderna e eficiente.
- Habilita e inicia o serviço chrony imediatamente, e também o coloca para iniciar automaticamente a cada boot.
- Força uma sincronização imediata do relógio do sistema com os servidores NTP configurados.
- Mostra os servidores NTP ativos, com detalhes (-v) sobre qualidade, latência e estratos.
- Exibe o status da sincronização do relógio, mostrando o servidor atual, o desvio do relógio e a precisão.

Configurar sudo para o grupo Sphi

```
# Criar o grupo
sudo groupadd sphi
# Adicionar usuários ao grupo
sudo usermod -aG sphi usuario1
sudo usermod -aG sphi usuario2
# Criar arquivo de regras
sudo nano /etc/sudoers.d/sphi
# Conteúdo do arquivo:
%sphi ALL=(ALL:ALL) ALL
# Ajustar permissões do arquivo
sudo chmod 440 /etc/sudoers.d/sphi
# Verificar sintaxe
sudo visudo -c -f /etc/sudoers.d/sphi
# Verificar membros do grupo
getent group sphi
```

- Cria o grupo sphi, caso não exista.
- Adiciona usuario1 e usuario2 ao grupo sphi, mantendo os grupos já existentes (-aG).
- Cria o arquivo /etc/sudoers.d/sphi com a regra %sphi ALL=(ALL:ALL) ALL, que concede todas as permissões de sudo para os membros do grupo sphi.
- Ajusta as permissões do arquivo para 440 (apenas leitura para root e grupo).
- Verifica a sintaxe do arquivo, garantindo que não há erros que possam quebrar o sudo.
- Exibe os membros do grupo sphi, confirmando quem foi adicionado.

Playbook DHCP

Configurar a interface do DHCP

```
# Criar/editar /etc/default/isc-dhcp-server
nano /etc/default/isc-dhcp-server
# Conteúdo:
INTERFACESv4="enp0s3"
INTERFACESv6=""
# Ajustar permissões
chmod 644 /etc/default/isc-dhcp-server
# Verificar
cat /etc/default/isc-dhcp-server
```

- Cria ou edita o arquivo /etc/default/isc-dhcp-server, onde ficam as configurações do serviço DHCP em Ubuntu.
- Define a variável INTERFACESv4="enp0s3", informando ao DHCP que ele deve escutar apenas na interface enp0s3 para requisições IPv4.
- Deixa INTERFACESv6="" vazio, desabilitando o DHCP para IPv6 (não há interface IPv6 configurada).
- Ajusta as permissões do arquivo para 644 (leitura para todos, escrita apenas para root).
- Exibe o conteúdo do arquivo, para confirmar que as configurações foram salvas corretamente.

Criar escopo DHCP para rede

```
# Criar/editar /etc/dhcp/dhcpd.conf
nano /etc/dhcp/dhcpd.conf
# Conteúdo:
subnet 192.168.0.0 netmask 255.255.255.0 {
  range 192.168.0.50 192.168.0.150;
  option routers 192.168.0.1;
}
# Validar sintaxe
dhcpd -t -cf /etc/dhcp/dhcpd.conf
# Reiniciar serviço
sudo systemctl restart isc-dhcp-server
# Verificar status
sudo systemctl status isc-dhcp-server --no-pager
```

- Cria ou edita o arquivo /etc/dhcp/dhcpd.conf, onde ficam as configurações do servidor DHCP.
- Define o escopo para a sub-rede 192.168.0.0/255.255.255.0, com faixa de IPs de 192.168.0.50 a 192.168.0.150 e gateway padrão 192.168.0.1.
- Valida a sintaxe do arquivo, garantindo que não há erros que possam impedir o serviço de iniciar.
- Reinicia o serviço DHCP para aplicar as novas configurações.
- Exibe o status do serviço, confirmando se ele está ativo e rodando corretamente.

Criar arquivo de configuração dhcpd.conf

```
# Criar/editar /etc/dhcp/dhcpd.conf
nano /etc/dhcp/dhcpd.conf
# Conteúdo:
ddns-update-style none;
authoritative;
# Sub-rede 192.168.0.0/24
subnet 192.168.0.0 netmask 255.255.255.0 {
  range 192.168.0.50 192.168.0.150;
  option routers 192.168.0.1;
  option domain-name-servers 8.8.8.8, 8.8.4.4;
  default-lease-time 600;
  max-lease-time 7200;
}
# Sub-rede 192.168.0.0/24 (VM-LX-1)
subnet 192.168.0.0 netmask 255.255.255.0 {
  range 192.168.0.50 192.168.0.150;
  option routers 192.168.0.1;
}
# Host fixo 1
host fixo1 {
  hardware ethernet 52:54:00:a1:b2:c3;
  fixed-address 192.168.0.101;
}
# Host fixo 2
host fixo2 {
  hardware ethernet 52:54:00:a1:b2:c4;
  fixed-address 192.168.0.102;
}
```

- Cria ou edita o arquivo /etc/dhcp/dhcpd.conf, onde ficam as configurações do servidor DHCP.
- Define ddns-update-style none; e authoritative; para desabilitar DDNS e tornar o servidor DHCP autoritativo na rede.
- Configura a sub-rede 192.168.0.0/255.255.255.0, com faixa de IPs de 192.168.0.50 a 192.168.0.150, gateway padrão 192.168.0.1, DNS 8.8.8.8 e 8.8.4.4, lease padrão de 600 segundos e lease máximo de 7200 segundos.
- Reserva os endereços 192.168.0.101 e 192.168.0.102 para os hosts fixos fixo1 e fixo2, identificados pelos endereços MAC 52:54:00:a1:b2:c3 e 52:54:00:a1:b2:c4.
- Ajusta as permissões do arquivo para 644 (leitura para todos, escrita apenas para root).
- Exibe o conteúdo do arquivo, para confirmar que as configurações foram salvas corretamente.

Ativação dos Bitwarden Firmas

```
# Criar diretórios
mkdir -p /opt/bitwarden/{mssql,data,logs}
# Ajustar permissões (o container roda como root)
chown -R root:root /opt/bitwarden
# Criar arquivo de variáveis de ambiente
cat > /opt/bitwarden/.env <<EOF
MSSQL_SA_PASSWORD=$(cat /opt/bitwarden/mssql/sa.pass)
ACCEPT_EULA=Y
MSSQL_PID=Express
EOF
```

- Criação dos diretórios para armazenar os dados do SQL Server (mssql), dados da aplicação (data) e logs (logs).
- Ajuste de permissões para que o container possa escrever nos diretórios montados.
- Gera uma senha forte para o usuário SA do SQL Server e salva em /opt/bitwarden/mssql/sa.pass.
- Cria o arquivo .env com as variáveis de ambiente necessárias para o container do SQL Server, incluindo a senha gerada, aceitação da EULA e versão Express.

Instalar pacotes internacionais

```
# Instalar pacotes
sudo apt-get install -y \
  language-pack-pt \
  language-pack-en \
  locales-all \
  tzdata \
  fonts-noto-cjk \
  fonts-noto-color-emoji
# Gerar locales
sudo locale-gen pt_BR.UTF-8 en_US.UTF-8
# Atualizar cache
sudo dpkg-reconfigure -f noninteractive locales
# Verificar
locale -a | grep -E "pt_BR|en_US"
```

- Instala os pacotes de idioma português (language-pack-pt) e inglês (language-pack-en), todos os locales adicionais (locales-all), fuso horário (tzdata), fontes Noto CJK (fonts-noto-cjk) e fontes de emoji colorido (fonts-noto-color-emoji).
- Gera os locales pt_BR.UTF-8 e en_US.UTF-8 no sistema.
- Atualiza o cache de locales do sistema, para que os novos idiomas sejam reconhecidos.
- Exibe os locales instalados, confirmando se pt_BR e en_US estão ativos.

Criar partição com discos adicionais

```
# Criar partições com fdisk (exemplo para /dev/sdb)
# Criar PV para cada disco
pvcreate /dev/sdb1 /dev/sdc1 /dev/sdd1
# Criar VG com todos os PVs
vgcreate vg_data /dev/sdb1 /dev/sdc1 /dev/sdd1
# Criar LV usando todo o espaço livre
lvcreate -l 100%FREE -n lv_data vg_data
# Formatar com ext4
mkfs.ext4 /dev/vg_data/lv_data
# Criar ponto de montagem
mkdir -p /mnt/data
# Montar
mount /dev/vg_data/lv_data /mnt/data
# Verificar
df -h /mnt/data
lsblk -f
```

- Cria três partições primárias (/dev/sdb1, /dev/sdc1, /dev/sdd1) com o tipo padrão Linux (Id 83).
- Inicializa os três volumes físicos (PV) para o LVM.
- Cria o volume group vg_data, agregando os três PVs.
- Cria o logical volume lv_data, ocupando 100% do espaço livre do VG.
- Formata o LV com o sistema de arquivos ext4.
- Cria o ponto de montagem /mnt/data.
- Monta o LV no ponto de montagem.
- Exibe o espaço livre em disco, confirmando que o volume está montado e ocupando o espaço esperado.
- Exibe os blocos, confirmando que o LV aparece como /dev/mapper/vg_data-lv_data e está montado em /mnt/data.

Criar Volumes Physicos (PV)

```
# Criar PV para cada disco
pvcreate /dev/sdb1 /dev/sdc1 /dev/sdd1
# Verificar
pvscan
pvdisplay /dev/sdb1 /dev/sdc1 /dev/sdd1
# Listar PVs
pvs -o pv_name,vg_name,pv_size,pv_free,pv_used
```

- Inicializa os três volumes físicos (PV) para o LVM.
- Exibe todos os PVs detectados no sistema.
- Mostra os detalhes dos três PVs.
- Lista os PVs em formato resumido, com nome, VG associado, tamanho, espaço livre e espaço usado.

Criar Lv Volume Group (VG)

```
# Criar VG com todos os PVs
vgcreate vg_data /dev/sdb1 /dev/sdc1 /dev/sdd1
# Verificar
vgscan
vgdisplay vg_data
# Listar VGs
vgs -o vg_name,pv_count,lv_count,vg_size,vg_free,vg_attr
```

- Cria o volume group vg_data, agregando os três PVs (/dev/sdb1, /dev/sdc1, /dev/sdd1).
- Exibe todos os VGs detectados no sistema.
- Mostra os detalhes do VG vg_data, incluindo PVs associados, LVs, tamanho total e espaço livre.
- Lista os VGs em formato resumido, com nome, quantidade de PVs e LVs, tamanho e espaço livre.

Criar Lv logical Volume (LV)

```
# Criar LV ocupando todo o espaço livre
lvcreate -l 100%FREE -n lv_data vg_data
# Criar LV com tamanho específico (exemplo: 50GB)
lvcreate -L 50G -n lv_data vg_data
# Verificar
lvscan
lvdisplay /dev/vg_data/lv_data
# Listar LVs
lvs -o lv_name,vg_name,lv_size,lv_path,lv_attr
```

- Cria o logical volume lv_data, ocupando todo o espaço livre do VG vg_data (100%FREE).
- Exibe todos os LVs detectados no sistema.
- Mostra os detalhes do LV lv_data, incluindo caminho, tamanho e status.
- Lista os LVs em formato resumido, com nome, VG associado, tamanho e caminho.

Formatar e Logical Volume

```
# Criar sistema de arquivos ext4 no LV
mkfs.ext4 /dev/vg_data/lv_data
# Verificar
blkid /dev/vg_data/lv_data
file -s /dev/vg_data/lv_data
```

- Formata o LV /dev/vg_data/lv_data com o sistema de arquivos ext4, criando a estrutura de arquivos e metadados.
- Exibe o UUID e o tipo do sistema de arquivos do LV, confirmando que foi formatado corretamente.
- Mostra o tipo do sistema de arquivos do LV, confirmando que é ext4.

Criar a Pasta de Mensagem

```
# Criar diretório
mkdir -p /var/spool/postfix/{public,maildrop}
# Ajustar permissões
chmod 730 /var/spool/postfix/maildrop
chmod 755 /var/spool/postfix/public
# Verificar
ls -ld /var/spool/postfix/maildrop
ls -ld /var/spool/postfix/public
```

- Cria os diretórios /var/spool/postfix/public e /var/spool/postfix/maildrop, onde o Postfix armazena mensagens em fila e mensagens prontas para entrega.
- Ajusta as permissões do diretório maildrop para 730 (rwx-wx---), permitindo que o usuário postfix escreva e o grupo execute, mas ninguém leia.
- Ajusta as permissões do diretório public para 755 (rwxr-xr-x), permitindo que todos leiam e o usuário postfix escreva.
- Exibe o conteúdo dos diretórios, para confirmar que foram criados e que as permissões foram aplicadas corretamente.

Adicionar ao /etc/hosts e Hostnames

```
# Editar /etc/hosts
nano /etc/hosts
# Adicionar:
127.0.0.1 localhost
127.0.1.1 nome-da-maquina
# Verificar
cat /etc/hosts
# Testar resolução
ping -c 1 nome-da-maquina
ping -c 1 localhost
# Ver hostnames
hostnamectl
hostnamectl status
```

- Adiciona 127.0.0.1 localhost e 127.0.1.1 nome-da-maquina ao /etc/hosts, garantindo que o hostname local seja resolvido corretamente.
- Exibe o conteúdo do /etc/hosts, para confirmar que as entradas foram adicionadas.
- Testa a resolução do hostname e do localhost, confirmando que o sistema consegue resolver os nomes.
- Exibe o hostname atual e o hostname completo (FQDN), para confirmar que foi configurado corretamente.

Instalar Docker Desktop

```
# Atualizar sistema
sudo apt update && sudo apt upgrade -y
# Instalar dependências
sudo apt install -y ca-certificates curl gnupg lsb-release
# Adicionar chave GPG oficial do Docker
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
# Adicionar repositório
echo "deb [arch=amd64 signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list
# Instalar Docker Engine
sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
# Adicionar usuário ao grupo docker
sudo usermod -aG docker $USER
# Ativar e iniciar
sudo systemctl enable --now docker
# Verificar instalação
docker --version
docker compose version
# Teste
sudo docker run hello-world
# Instalar Docker Desktop
wget -O /tmp/docker-desktop.deb https://desktop.docker.com/linux/main/amd64/docker-desktop-amd64.deb
sudo apt install -y /tmp/docker-desktop.deb
# Ativar
systemctl --user enable docker-desktop
# Verificar
docker desktop version
```

Criar Volume Group (VG)

```
# Criar VG com todos os PVs
vgcreate vg_data /dev/sdb1 /dev/sdc1 /dev/sdd1
# Verificar
vgscan
vgdisplay vg_data
# Listar VGs
vgs -o vg_name,pv_count,lv_count,vg_size,vg_free,vg_attr
```

- Cria o volume group vg_data, agregando os três PVs (/dev/sdb1, /dev/sdc1, /dev/sdd1).
- Exibe todos os VGs detectados no sistema.
- Mostra os detalhes do VG vg_data, incluindo PVs associados, LVs, tamanho total e espaço livre.
- Lista os VGs em formato resumido, com nome, quantidade de PVs e LVs, tamanho e espaço livre.

Criar Lv logical Volume (LV)

```
# Criar LV ocupando todo o espaço livre
lvcreate -l 100%FREE -n lv_data vg_data
# Criar LV com tamanho específico (exemplo: 50GB)
lvcreate -L 50G -n lv_data vg_data
# Verificar
lvscan
lvdisplay /dev/vg_data/lv_data
# Listar LVs
lvs -o lv_name,vg_name,lv_size,lv_path,lv_attr
```

- Cria o logical volume lv_data, ocupando todo o espaço livre do VG vg_data (100%FREE).
- Exibe todos os LVs detectados no sistema.
- Mostra os detalhes do LV lv_data, incluindo caminho, tamanho e status.
- Lista os LVs em formato resumido, com nome, VG associado, tamanho e caminho.

Formatar e Logical Volume

```
# Criar sistema de arquivos ext4 no LV
mkfs.ext4 /dev/vg_data/lv_data
# Verificar
blkid /dev/vg_data/lv_data
file -s /dev/vg_data/lv_data
```

- Formata o LV /dev/vg_data/lv_data com o sistema de arquivos ext4, criando a estrutura de arquivos e metadados.
- Exibe o UUID e o tipo do sistema de arquivos do LV, confirmando que foi formatado corretamente.
- Mostra o tipo do sistema de arquivos do LV, confirmando que é ext4.

Criar a Pasta de Mensagem

```
# Criar diretório
mkdir -p /var/spool/postfix/{public,maildrop}
# Ajustar permissões
chmod 730 /var/spool/postfix/maildrop
chmod 755 /var/spool/postfix/public
# Verificar
ls -ld /var/spool/postfix/maildrop
ls -ld /var/spool/postfix/public
```

- Cria os diretórios /var/spool/postfix/public e /var/spool/postfix/maildrop, onde o Postfix armazena mensagens em fila e mensagens prontas para entrega.
- Ajusta as permissões do diretório maildrop para 730 (rwx-wx---), permitindo que o usuário postfix escreva e o grupo execute, mas ninguém leia.
- Ajusta as permissões do diretório public para 755 (rwxr-xr-x), permitindo que todos leiam e o usuário postfix escreva.
- Exibe o conteúdo dos diretórios, para confirmar que foram criados e que as permissões foram aplicadas corretamente.

Adicionar ao /etc/hosts e Hostnames

```
# Editar /etc/hosts
nano /etc/hosts
# Adicionar:
127.0.0.1 localhost
127.0.1.1 nome-da-maquina
# Verificar
cat /etc/hosts
# Testar resolução
ping -c 1 nome-da-maquina
ping -c 1 localhost
# Ver hostnames
hostnamectl
hostnamectl status
```

- Adiciona 127.0.0.1 localhost e 127.0.1.1 nome-da-maquina ao /etc/hosts, garantindo que o hostname local seja resolvido corretamente.
- Exibe o conteúdo do /etc/hosts, para confirmar que as entradas foram adicionadas.
- Testa a resolução do hostname e do localhost, confirmando que o sistema consegue resolver os nomes.
- Exibe o hostname atual e o hostname completo (FQDN), para confirmar que foi configurado corretamente.

Instalar Docker Desktop

```
# Atualizar sistema
sudo apt update && sudo apt upgrade -y
# Instalar dependências
sudo apt install -y ca-certificates curl gnupg lsb-release
# Adicionar chave GPG oficial do Docker
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
# Adicionar repositório
echo "deb [arch=amd64 signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list
# Instalar Docker Engine
sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
# Adicionar usuário ao grupo docker
sudo usermod -aG docker $USER
# Ativar e iniciar
sudo systemctl enable --now docker
# Verificar instalação
docker --version
docker compose version
# Teste
sudo docker run hello-world
# Instalar Docker Desktop
wget -O /tmp/docker-desktop.deb https://desktop.docker.com/linux/main/amd64/docker-desktop-amd64.deb
sudo apt install -y /tmp/docker-desktop.deb
# Ativar
systemctl --user enable docker-desktop
# Verificar
docker desktop version
```

Instalar pacotes internacionais

```
# Instalar pacotes
sudo apt-get install -y \
  language-pack-pt \
  language-pack-en \
  locales-all \
  tzdata \
  fonts-noto-cjk \
  fonts-noto-color-emoji
# Gerar locales
sudo locale-gen pt_BR.UTF-8 en_US.UTF-8
# Atualizar cache
sudo dpkg-reconfigure -f noninteractive locales
# Verificar
locale -a | grep -E "pt_BR|en_US"
```

- Instala os pacotes de idioma português (language-pack-pt) e inglês (language-pack-en), todos os locales adicionais (locales-all), fuso horário (tzdata), fontes Noto CJK (fonts-noto-cjk) e fontes de emoji colorido (fonts-noto-color-emoji).
- Gera os locales pt_BR.UTF-8 e en_US.UTF-8 no sistema.
- Atualiza o cache de locales do sistema, para que os novos idiomas sejam reconhecidos.
- Exibe os locales instalados, confirmando se pt_BR e en_US estão ativos.

Criar partição com discos adicionais

```
# Criar partições com fdisk (exemplo para /dev/sdb)
# Criar PV para cada disco
pvcreate /dev/sdb1 /dev/sdc1 /dev/sdd1
# Criar VG com todos os PVs
vgcreate vg_data /dev/sdb1 /dev/sdc1 /dev/sdd1
# Criar LV usando todo o espaço livre
lvcreate -l 100%FREE -n lv_data vg_data
# Formatar com ext4
mkfs.ext4 /dev/vg_data/lv_data
# Criar ponto de montagem
mkdir -p /mnt/data
# Montar
mount /dev/vg_data/lv_data /mnt/data
# Verificar
df -h /mnt/data
lsblk -f
```

- Cria três partições primárias (/dev/sdb1, /dev/sdc1, /dev/sdd1) com o tipo padrão Linux (Id 83).
- Inicializa os três volumes físicos (PV) para o LVM.
- Cria o volume group vg_data, agregando os três PVs.
- Cria o logical volume lv_data, ocupando 100% do espaço livre do VG.
- Formata o LV com o sistema de arquivos ext4.
- Cria o ponto de montagem /mnt/data.
- Monta o LV no ponto de montagem.
- Exibe o espaço livre em disco, confirmando que o volume está montado e ocupando o espaço esperado.
- Exibe os blocos, confirmando que o LV aparece como /dev/mapper/vg_data-lv_data e está montado em /mnt/data.

Criar Volumes Physicos (PV)

```
# Criar PV para cada disco
pvcreate /dev/sdb1 /dev/sdc1 /dev/sdd1
# Verificar
pvscan
pvdisplay /dev/sdb1 /dev/sdc1 /dev/sdd1
# Listar PVs
pvs -o pv_name,vg_name,pv_size,pv_free,pv_used
```

- Inicializa os três volumes físicos (PV) para o LVM.
- Exibe todos os PVs detectados no sistema.
- Mostra os detalhes dos três PVs.
- Lista os PVs em formato resumido, com nome, VG associado, tamanho, espaço livre e espaço usado.

Criar Lv Volume Group (VG)

```
# Criar VG com todos os PVs
vgcreate vg_data /dev/sdb1 /dev/sdc1 /dev/sdd1
# Verificar
vgscan
vgdisplay vg_data
# Listar VGs
vgs -o vg_name,pv_count,lv_count,vg_size,vg_free,vg_attr
```

- Cria o volume group vg_data, agregando os três PVs (/dev/sdb1, /dev/sdc1, /dev/sdd1).
- Exibe todos os VGs detectados