

Configuração de Cluster com Slurm no Ubuntu LTS

Guia sobre como configurar o cluster gpu no Ubuntu 22.04 usando slurm (com cgroups).

0. Suposições

- masternode 100.xx.100.xx (**Ex:** 10.211.55.12)
- workernode 200.xx.200.xx (**Ex:** 10.211.55.13)
- masternode FQDN = masternode.master.local
- workernode FQDN = workernode.worker.local

Obs: Pode ser interessante instalar o pacote `iptutils-ping` através do comando `sudo apt install iptutils-ping`

0.1 No masternode

1. Para definir o nome do host temporariamente (até a próxima reinicialização), utilize o comando `hostname` seguido do novo nome desejado:

```
1 | sudo hostnamectl set-hostname workernode
```

2. Para definir o nome do host permanentemente, você precisa editar dois arquivos:

```
1 | sudo nano /etc/hostname
```

Edite o conteúdo do arquivo para o novo nome do host desejado. Por exemplo, para definir como "masternode":

```
1 | masternode
```

Salve o arquivo e feche o editor.

3. `/etc/hosts`: Este arquivo armazena o mapeamento entre nomes de host e endereços IP. Abra-o em um editor de texto:

```
1 | sudo nano /etc/hosts
```

Edite o conteúdo do arquivo para o novo nome do host desejado. Por exemplo, para definir como "masternode":

```
1 | 127.0.0.1 localhost masternode
2 | 127.0.1.1 masternode
3 | 100.xx.100.xx masternode
4 | 200.xx.200.xx workernode
```

Salve o arquivo e feche o editor.

4. Atualize o nome do host:

Para que as alterações permanentes entrem em vigor, você precisa atualizar o nome do host na memória:

```
1 | sudo reboot
```

0.2 No workernode

1. Para definir o nome do host temporariamente (até a próxima reinicialização), utilize o comando `hostname` seguido do novo nome desejado:

```
1 | sudo hostnamectl set-hostname workernode
```

2. Para definir o nome do host permanentemente, você precisa editar dois arquivos:

```
1 | sudo nano /etc/hostname
```

Edite o conteúdo do arquivo para o novo nome do host desejado. Por exemplo, para definir como "masternode":

```
1 | workernode
```

Salve o arquivo e feche o editor.

3. `/etc/hosts`: Este arquivo armazena o mapeamento entre nomes de host e endereços IP. Abra-o em um editor de texto:

```
1 | sudo nano /etc/hosts
```

Edite o conteúdo do arquivo para o novo nome do host desejado. Por exemplo, para definir como "masternode":

```
1 | 127.0.0.1 localhost workernode
2 | 127.0.1.1 workernode
3 | 100.xx.100.xx masternode
4 | 200.xx.200.xx workernode
```

Salve o arquivo e feche o editor.

4. Atualize o nome do host:

Para que as alterações permanentes entrem em vigor, você precisa atualizar o nome do host na memória:

```
1 | sudo reboot
```

1. Instale drivers nvidia em todos os nodes

Atualize o sistema operacional

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

Remova a instalação anterior da NVIDIA

```
1 | sudo apt autoremove nvidia* --purge
```

Execute o seguinte comando:

```
1 | ubuntu-drivers devices
```

- Este comando listará os drivers NVIDIA disponíveis para sua placa de vídeo.
- Identifique o driver mais recente na lista.
- Execute o seguinte comando, substituindo `x` pelo número do driver (por exemplo, `nvidia-driver-550`):

```
1 | sudo apt install nvidia-driver-x
```

- Siga as instruções na tela para concluir a instalação.

Reinicie o SO e verifique a instalação:

```
1 | nvidia-smi
```

Instale o CUDA toolkit

```
1 | sudo apt install nvidia-cuda-toolkit
```

Verifique a instalação do CUDA

```
1 | nvcc --version
```

1.1 Alternativamente para instalar o Cuda 12.5 com cuda drivers

Atualize o sistema operacional

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

Remova a instalação anterior da NVIDIA

```
1 | sudo apt autoremove nvidia* --purge
```

Monte o seguinte comando:

```
1 | wget
  https://developer.download.nvidia.com/compute/cuda/<release>/local_installers/cuda_<re
  lease>_<version>_linux.run
2 | sh cuda_<release>_<version>_linux.run -m=kernel-open
```

Obs: Verifique a versão do Cuda mais recente em https://developer.nvidia.com/cuda-downloads?target_os=Linux&target_arch=x86_64&Distribution=Ubuntu

Por exemplo:

```
1 | wget
  https://developer.download.nvidia.com/compute/cuda/12.5.0/local_installers/cuda_12.5.0
  _555.42.02_linux.run
2 | sudo sh cuda_12.5.0_555.42.02_linux.run -m=kernel-open
```

Reinicie o SO e verifique a instalação:

```
1 | nvidia-smi
2 | nvcc --version
```

2. Configure o ssh sem senha

1. Instalação do servidor SSH e Firewall

No masternode e no workernode:

```
1 | sudo apt install openssh-server ufw
2 | sudo ufw enable
3 | sudo ufw allow ssh
```

```
1 |
```

2. Gerar as chaves SSH na máquina master

```
1 | ssh-keygen -t rsa -b 4096
```

Quando solicitado a fornecer um caminho para salvar a chave, pressione **Enter** para aceitar o local padrão (`~/.ssh/id_rsa`).

Quando solicitado a fornecer uma senha (passphrase), simplesmente pressione **Enter** para deixá-la vazia (sem senha).

3. Copiar a chave pública para a máquina slave

```
1 | ssh-copy-id usuario@200.xx.200.xx
```

- **usuario**: Substitua pelo nome de usuário na máquina slave.
- **200.xx.200.xx**: Substitua pelo endereço IP ou hostname da máquina worker.

Se solicitado, insira a senha do usuário na máquina slave para completar a cópia da chave.

4. Testar a configuração SSH sem senha

Ainda na máquina master, tente conectar-se à máquina slave sem fornecer uma senha:

```
1 | ssh usuario@200.xx.200.xx
```

Você deve ser capaz de acessar a máquina slave sem ser solicitado a fornecer uma senha.

5. Configuração adicional (opcional)

Para garantir que a conexão SSH funcione corretamente e para facilitar o uso futuro, você pode adicionar um bloco de configuração ao arquivo `~/.ssh/config` na máquina master. Edite o arquivo ou crie-o se não existir:

```
1 | nano ~/.ssh/config
```

Adicione a seguinte configuração, substituindo os valores apropriados:

```
1 | Host workernode
2 |     HostName 200.xx.200.xx
3 |     User usuario
4 |     IdentityFile ~/.ssh/id_rsa
```

Isso permite que você se conecte à máquina slave simplesmente usando o comando:

```
1 | ssh workernode
```

6. Verificação de permissões

Certifique-se de que as permissões dos arquivos de chave SSH e diretórios são corretas:

- Assegure-se de que o diretório `~/.ssh` tenha permissão 700:

```
1 | chmod 700 ~/.ssh
```

- Assegure-se de que os arquivos de chave privada `~/.ssh/id_rsa` tenham permissão 600 no `masternode`:

```
1 | chmod 600 ~/.ssh/id_rsa
```

- o Assegure-se de que os arquivos de chave pública `~/.ssh/id_rsa.pub` e `~/.ssh/authorized_keys` (na workernode) tenham permissão 644:

```
1 | chmod 644 ~/.ssh/id_rsa.pub
2 | chmod 644 ~/.ssh/authorized_keys
```

3. Sincronize GID/UIDs

3.1 Crie usuários munge e slurm:

Nos nós masternode e workernode:

```
1 | sudo adduser -u 1111 munge --disabled-password --gecos ""
2 | sudo adduser -u 1121 slurm --disabled-password --gecos ""
```

E atualize as permissões de acordo com suas preferências.

4. Sincronize a hora

Em todos os nós, instale o NTP e defina o fuso horário para UTC.

```
1 | sudo apt-get install ntp
2 | sudo dpkg-reconfigure tzdata
3 | Selecione America/Sao_Paulo
```

4.1 NTP Server

Escolha um nó como servidor de hora principal (**sugestão**: nó master) e configure-o para funcionar corretamente mesmo que não esteja ligado à Internet.

```
1 | sudo nano /etc/ntp.conf
```

Adicione o seguinte para fornecer a sua hora local atual como predefinição, caso perca temporariamente (ou permanentemente) a ligação à Internet:

```
1 | server 127.127.1.0
2 | fudge 127.127.1.0 stratum 10
```

Reinicie o NTP

```
1 | sudo /etc/init.d/ntp restart
```

4.2 NTP Client

Em todos os restantes nós do seu cluster, configure-os para sincronizarem os relógios com o nó que foi designado como o servidor de hora principal do cluster.

```
1 | sudo nano /etc/ntp.conf
```

Adicione o seguinte :

```
1 | server <main time server> iburst
```

em que `<main time server>` é o endereço IP do nó designado como o servidor de hora principal.

Remova as seguintes linhas do arquivo comentando-as com um carácter `#` no início da linha:

```
1 | # Use servers from the NTP Pool Project. Approved by Ubuntu Technical Board
2 | # on 2011-02-08 (LP: #104525). See http://www.pool.ntp.org/join.html for
3 | # more information.
4 | #server 0.ubuntu.pool.ntp.org
5 | #server 1.ubuntu.pool.ntp.org
6 | #server 2.ubuntu.pool.ntp.org
7 | #server 3.ubuntu.pool.ntp.org
8 |
9 | # Use Ubuntu's ntp server as a fallback.
10 | #server ntp.ubuntu.com
```

Reinicie o NTP

```
1 | sudo /etc/init.d/ntp restart
```

Verificar a conectividade com o servidor de hora principal:

```
1 | ntpq -c lpeer
```

#####

5. Configure o NFS

Para que o SLURM funcione corretamente, deve haver um local de armazenamento presente em todos os computadores do cluster com os mesmos arquivos usados para os trabalhos. Todos os computadores do cluster devem ser capazes de ler e gravar nesse diretório. Uma maneira de fazer isso é com o NFS.

NFS, ou Network File System, é um protocolo de sistema de arquivos distribuído que permite montar diretórios remotos no seu servidor. Isso permite gerenciar o espaço de armazenamento em um local diferente e gravar nesse espaço a partir de vários clientes. O NFS fornece uma forma relativamente rápida e fácil de aceder a sistemas remotos através de uma rede e funciona bem em situações em que os recursos partilhados serão acedidos regularmente.

5.1 Master node:

```
1 sudo apt update
2 sudo apt install nfs-kernel-server
3 sudo mkdir /storage -p
4 sudo chown usuario:grupo /storage/
5 sudo nano /etc/exports
6 /storage 200.xx.200.xx(rw,sync,no_root_squash,no_subtree_check)
7 sudo systemctl restart nfs-kernel-server
8 sudo ufw allow from 200.xx.200.xx to any port nfs
```

Substitua `200.xx.200.xx` pelo IP do workenode.

Substitua `usuario:grupo` por um usuário e grupo existente no masternode.

Obs: Note que a `linha 6` exemplifica apenas um único nó. Devem existir uma linha para cada nó que acessará o volume, exceto para o nó hospedeiro do volume.

5.2 Worker node:

```
1 sudo apt update
2 sudo apt install nfs-common
3 sudo mkdir -p /storage
4 sudo mount 100.xx.100.xx:/storage /storage
5 echo 100.xx.100.xx:/storage /storage nfs auto,timeo=14,intr 0 0 | sudo tee -a
  /etc/fstab
6 sudo chown usuario:grupo /storage/
```

Substitua `100.xx.100.xx` pelo IP do masternode.

Substitua `usuario:grupo` por um usuário e grupo existente no workenode.

6. Configure o MUNGE

MUNGE (MUNGE Uid 'N' Gid Emporium) é uma ferramenta de autenticação que cria e valida credenciais de segurança para processos distribuídos.

MUNGE é utilizado para autenticar mensagens entre processos em clusters de computadores, garantindo que a comunicação entre diferentes nós do cluster seja segura e confiável. Ele é especialmente útil em ambientes de computação de alto desempenho (HPC) onde a segurança e a integridade da comunicação entre nós são críticas.

6.1 Master node:


```
1 | sudo apt install libmunge-dev libmunge2 munge -y
2 | sudo systemctl enable munge
3 | sudo systemctl start munge
4 | munge -n | unmunge | grep STATUS
5 | sudo cp /etc/munge/munge.key /storage/
6 | sudo chown munge /storage/munge.key
7 | sudo chmod 400 /storage/munge.key
```

Você deverá obter `STATUS: Success (0)` ao executar a instrução 4.

6.2 Worker node:

```
1 | sudo apt install libmunge-dev libmunge2 munge
2 | sudo cp /storage/munge.key /etc/munge/munge.key
3 | sudo systemctl enable munge
4 | sudo systemctl start munge
5 | munge -n | unmunge | grep STATUS
```

Você deverá obter `STATUS: Success (0)` ao executar a instrução 5.

7. Configure o Slurm

SLURM (Simple Linux Utility for Resource Management) é um sistema de gerenciamento de recursos e fila de jobs (workload manager) utilizado em ambientes de computação de alto desempenho (HPC).

SLURM gerencia e aloca recursos de computação em clusters, permitindo que os usuários enviem, agendem e monitorem jobs (tarefas de computação) de forma eficiente. Ele cuida do balanceamento de carga, agendamento de jobs, controle de acesso, e monitoramento de recursos, facilitando o uso otimizado do cluster e garantindo que os recursos sejam utilizados de forma justa e eficiente.

7.1 Configure o DB para o Slurm

Os procedimentos deverão ser realizados no `masternode`.

1. Instale o cliente para o `git`:

```
1 | sudo apt install git
```

2. Clone este repositório com ficheiros de configuração e de serviço:

```
1 | cd /storage
2 | git clone https://github.com/lopentuska/slurm_ubuntu_gpu_cluster
```

3. Instalar pré-requisitos para BD:

```

1 sudo apt install python3 gcc make openssl ruby ruby-dev libpam0g-dev libmariadb-dev
  mariadb-server build-essential libssl-dev numactl hwloc libmunge-dev man2html lua5.3 -
  y
2 sudo gem install fpm
3 sudo systemctl enable mysql
4 sudo systemctl start mysql
5 sudo systemctl status mysql | grep Status:

```

Você deverá ver `Status: "Taking your SQL requests now..."`.

4. Criando o usuário e o BD para o Slurm:

```

1 sudo mysql -u root

```

```

1 create database slurm_acct_db;
2 create user 'slurm'@'localhost';
3 set password for 'slurm'@'localhost' = password('senha_para_usuario_slurm');
4 grant usage on *.* to 'slurm'@'localhost';
5 grant all privileges on slurm_acct_db.* to 'slurm'@'localhost';
6 flush privileges;
7 exit

```

Obs: Modifique a `senha_para_usuario_slurm` para a senha desejada para o usuário `slurm`.

7.2 Configure o Slurm

7.2.1 Configuração do Slurm no `masternode`

7.2.1.1 Criar arquivo de instalação

Você deve verificar a página de [download](#) do slurm e instalar a versão mais recente.

```

1 cd /storage
2 wget https://download.schedmd.com/slurm/slurm-24.05.0.tar.bz2
3 tar xvjf slurm-24.05.0.tar.bz2
4 cd slurm-24.05.0/
5 ./configure --prefix=/tmp/slurm-build --sysconfdir=/etc/slurm --enable-pam --with-
  pam_dir=/lib/x86_64-linux-gnu/security/ --without-shared-libslurm
6 make
7 make contrib
8 make install
9 cd ..

```

Em caso de falhas: Em algumas distribuições, o `pam_dir` pode ser `/lib/security/` ou `/usr/lib/security/`. Para localizar o diretório correto, execute `find /lib -name '*pam*.so'` ou `find /usr/lib -name '*pam*.so'`. O diretório desejado conterá as bibliotecas `pam_debug.so`, `pam_selinux.so` e `pam_group.so`. O diretório pode ter uma estrutura dada por `/usr/lib/ARQUITETURA-linux-gnu/security/`.

7.2.1.2 Instalar o Slurm

1. Criando e instalando o pacote do Slurm

```
1 sudo fpm -s dir -t deb -v 1.0 -n slurm-24.05.0 --prefix=/usr -C /tmp/slurm-build/ .
2 sudo dpkg -i slurm-24.05.0_1.0_arm64.deb
```

OBS: Observe a saída do comando da linha 1. Uma mensagem como `Created package {path=>"slurm-24.05.0_1.0_arm64.deb"}` será apresentada. O `path` apresentado na mensagem que deve ser utilizado na linha 2. e na instalação no `workernode`.

2. Criando os diretórios:

```
1 sudo mkdir -p /etc/slurm /etc/slurm/prolog.d /etc/slurm/epilog.d /var/spool/slurm/ctld
  /var/spool/slurm/d /var/log/slurm
2 sudo chown slurm /var/spool/slurm/ctld /var/spool/slurm/d /var/log/slurm
```

3. Copiar serviços slurm:

```
1 sudo cp /storage/slurm_ubuntu_gpu_cluster/configs_services/slurmdbd.service
  /etc/systemd/system/
2 sudo cp /storage/slurm_ubuntu_gpu_cluster/configs_services/slurmctld.service
  /etc/systemd/system/
```

4. Copiar a configuração do slurm DB:

```
1 sudo cp /storage/slurm_ubuntu_gpu_cluster/configs_services/slurmdbd.conf /etc/slurm/
2 sudo chmod 600 /etc/slurm/slurmdbd.conf
3 sudo chown slurm /etc/slurm/slurmdbd.conf
```

Edite o arquivo `/etc/slurm/slurmdbd.conf` para definir a senha do BD do usuário `slurm`:

```
1 StoragePass=senha_para_usuario_slurm
```

5. Portas abertas para comunicação com o slurm:

```
1 sudo ufw allow from any to any port 6817
2 sudo ufw allow from any to any port 6818
```

6. Inicie os serviços slurm:

```
1 sudo systemctl daemon-reload
2 sudo systemctl enable slurmdbd
3 sudo systemctl start slurmdbd
4 sudo systemctl enable slurmctld
5 sudo systemctl start slurmctld
```

OBS: O serviço ainda não está configurado, então `sudo systemctl status slurmd` retornará **falha**. Caso deseje testar no `masternode`, execute o passo seguinte ou termine de configurar o `workernode`.

7. Se o `masternode` for um nó de computação (`worker`) [aconselho a incluí-lo, por facilidade, com recursos limitados]:

```
1 sudo cp /storage/slurm_ubuntu_gpu_cluster/configs_services/slurmd.service
  /etc/systemd/system/
2 sudo systemctl enable slurmd
3 sudo systemctl start slurmd
```

###

7.2.2 Configuração do Slurm no `workernode`

7.2.2.1 Instalação do Slurm

Você deve verificar a página de [download](#) do slurm e instalar a versão mais recente.

```
1 cd /storage
2 sudo dpkg -i slurm-24.05.0_1.0_arm64.deb
3 sudo cp /storage/slurm_ubuntu_gpu_cluster/configs_services/slurmd.service
  /etc/systemd/system
```

7.2.2.2 Portas abertas para comunicação com o slurm:

```
1 sudo ufw allow from any to any port 6817
2 sudo ufw allow from any to any port 6818
```

```
1 sudo systemctl enable slurmd
2 sudo systemctl start slurmd
```

7.2.2.3 Configurar o Slurm:

Em `/storage/slurm_ubuntu_gpu_cluster/configs_services/slurm.conf`, altere:

```
1 ControlMachine=masternode.master.local # use seu FQDN
2 ControlAddr=100.xx.100.xx # use o IP do masternode
```

Use `sudo slurmd -C` para imprimir as especificações da máquina. Você deve copiar as especificações de todas as máquinas no arquivo `slurm.conf` e modificá-lo.

Exemplo de como deve ficar em seu arquivo de configuração:

```
1 NodeName=workernode CPUs=2 Boards=1 SocketsPerBoard=2 CoresPerSocket=1
  ThreadsPerCore=1 RealMemory=1967
```

Depois que você terminar de editar o `slurm.conf`:

```
1 | sudo cp /storage/slurm_ubuntu_gpu_cluster/configs_services/slurm.conf /storage/
```

Edite o arquivo `/storage/slurm_ubuntu_gpu_cluster/configs_services/gres.conf`.

```
1 | NodeName=masternode Name=gpu File=/dev/nvidia0
2 | NodeName=workernode Name=gpu File=/dev/nvidia0
```

Você pode usar o `nvidia-smi` para descobrir o número que deve ser usado em vez de `0` em `nvidia0`. Você o encontrará à esquerda do nome da GPU.

Caso não tenha GPUs, comente a linha `GresTypes=gpu` no arquivo

`/storage/slurm_ubuntu_gpu_cluster/configs_services/slurm.conf`.

No `workernode`, crie o diretório slurm: `sudo mkdir /etc/slurm/`

Copie os arquivos `.conf` (exceto `slurmdbd.conf`) em **todas as máquinas (workers e master node)**.

```
1 | sudo cp /storage/slurm_ubuntu_gpu_cluster/configs_services/cgroup* /etc/slurm/
2 | sudo cp /storage/slurm_ubuntu_gpu_cluster/configs_services/slurm.conf /etc/slurm/
3 | sudo cp /storage/slurm_ubuntu_gpu_cluster/configs_services/gres.conf /etc/slurm/
```

Esse diretório também deve ser criado nos `workers`:

```
1 | sudo mkdir -p /var/spool/slurm/d
2 | sudo chown slurm /var/spool/slurm/d
```

7.2.2.4 Configurar cgroups

```
1 | sudo nano /etc/default/grub
```

Para implementar as limitações de memória dos trabalhos e usuários do SLURM. Defina cgroups de memória em todos os trabalhadores com a adição da linha abaixo no arquivo `/etc/default/grub`:

```
1 | GRUB_CMDLINE_LINUX_DEFAULT="cgroup_enable=memory systemd.unified_cgroup_hierarchy=0"
```

Então

```
1 | sudo update-grub
```

7.3 Iniciar Slurm

Reinicie as máquinas (masternode e nós de trabalho).

7.3.1 Em masternode

```
1 | sudo systemctl restart slurmd
2 | sudo systemctl restart slurmdbd
```

Se o masternode for um nó de computação:

```
1 | sudo systemctl restart slurmd
```

7.3.2 Em worknode:

```
1 | sudo systemctl restart slurmd
```

Finalmente, nem **todos** os nós, execute:

```
1 | sudo apt update
2 | sudo apt upgrade
3 | sudo apt autoremove
```

8. Logs

Se algo não funcionar, você poderá encontrar os registros de `slurmctld`, `slurmdbd` e `slurmd` em `/var/log/slurm/`.

9. Scripts

Também adicionei um script simples para verificar se o `slurm` funciona, que executaria `srun hostname`, que basicamente imprimiria o nó no qual o trabalho foi iniciado.

Você precisará mover o arquivo para o diretório `/storage`.

Dentro do script, altere:

```
partition,
```

```
nodelist (escolha em qual nó será executado),
```

Em seguida, você pode executar o script com:

```
sbatch script_slurm_hostname.sh
```

9.2 script_slurm_hostname.sh

```
1  #!/bin/bash
2  #SBATCH --job-name=script_slurm_hostname
3  #SBATCH --partition=debug
4  #SBATCH --odelist=workernode
5  #SBATCH --nodes=1
6  #SBATCH --ntasks=1
7  #SBATCH --cpus-per-task=2
8  #SBATCH --gres=gpu:1
9  #SBATCH --time=00:00:30
10 srun hostname
```

10. Gerenciando usuários

10.1 Adicionando usuários

Utilize o script `add_user.sh` para adicionar usuários novos:

```
1  #!/bin/bash
2
3  # Verifica se o script foi executado com privilégios de superusuário
4  if [ "$(id -u)" -ne 0 ]; then
5      echo "Este script precisa ser executado como root."
6      exit 1
7  fi
8
9  # Verifica se o nome do usuário foi fornecido como argumento
10 if [ -z "$1" ]; then
11     echo "Uso: $0 nome_do_usuario [data_de_suspensao] [grupo]"
12     echo "    data_de_suspensao deve ser fornecida no formato AAAA-MM-DD"
13     echo "    grupo é opcional, o padrão é 'lamemo'"
14     exit 1
15 fi
16
17 # Atribui o nome do usuário à variável USERNAME
18 USERNAME=$1
19
20 # Verifica se a data de suspensão foi fornecida como argumento
21 SUSPENSION_DATE=$2
22
23 # Atribui o nome do grupo à variável GROUP ou usa 'lamemo' como padrão
24 GROUP=${3:-lamemo}
25
26 # Gera uma senha aleatória de 12 caracteres
27 PASSWORD=$(openssl rand -base64 12)
28
29 # Verifica se o grupo especificado existe e cria se necessário
30 if ! getent group "$GROUP" > /dev/null; then
```

```

31     groupadd "$GROUP"
32     echo "Grupo '$GROUP' criado."
33 fi
34
35 # Criando a pasta home para os usuarios
36 sudo mkdir -p /storage/home/
37
38 # Cria o usuário com o diretório home especificado e adiciona ao grupo especificado
39 useradd -m -d /storage/home/"$USERNAME" -s /bin/bash -g "$GROUP" "$USERNAME"
40
41 # Define a senha para o usuário
42 echo "$USERNAME:$PASSWORD" | chpasswd
43
44 # Exige que o usuário mude a senha no primeiro login
45 chage -d 0 "$USERNAME"
46
47 # Se uma data de suspensão foi fornecida, define a data de expiração da conta
48 if [ -n "$SUSPENSION_DATE" ]; then
49     chage -E "$SUSPENSION_DATE" "$USERNAME"
50     echo "A conta do usuário '$USERNAME' será suspensa em $SUSPENSION_DATE."
51 fi
52
53 # Exibe a senha gerada
54 echo "Usuário '$USERNAME' foi criado com sucesso."
55 echo "Senha temporária: $PASSWORD"
56 echo "Grupo: $GROUP"

```

Como Executar o Script

Salve o script em um arquivo chamado `add_user.sh`, dê permissão de execução e execute-o com os argumentos apropriados:

```

1  chmod +x add_user.sh
2  sudo ./add_user.sh nome_do_usuario [data_de_suspensao] [grupo]

```

Exemplos:

1. Para criar um usuário chamado `joao` que será suspenso em `2024-12-31` e adicionar ao grupo `alunos-hpc-2024-4`:

```

1  sudo ./add_user.sh joao 2024-12-31 alunos-hpc-2024-4

```

1. Para criar um usuário chamado `maria` sem data de suspensão e usar o grupo padrão `lamemo`:

```

1  sudo ./add_user.sh maria

```

1. Para criar um usuário chamado `ana` sem data de suspensão e adicionar ao grupo `alunos`:

```

1  sudo ./add_user.sh ana "" alunos

```


Este script atualizado permite que você forneça um grupo opcional para o usuário, criando e usando o grupo especificado ou padrão conforme necessário.

10.2 Removendo usuários com contas expiradas

Utilize o script `delete_expired_users.sh` para adicionar usuários novos:

```
1  #!/bin/bash
2
3  # Verifica se o script foi executado com privilégios de superusuário
4  if [ "$(id -u)" -ne 0 ]; then
5      echo "Este script precisa ser executado como root."
6      exit 1
7  fi
8
9  # Verifica se o número de dias foi fornecido como argumento
10 if [ -z "$1" ]; then
11     echo "Uso: $0 numero_de_dias"
12     exit 1
13 fi
14
15 # Atribui o número de dias à variável DAYS
16 DAYS=$1
17
18 # Data atual em segundos desde Epoch
19 CURRENT_DATE=$(date +%s)
20
21 # Obtém a lista de todos os usuários e verifica se estão no grupo lamemo
22 USERS=$(getent passwd | awk -F: '{print $1}')
23
24 for USER in $USERS; do
25     if id -nG "$USER" | grep -qw "lamemo"; then
26         # Obtém a data de expiração da conta do usuário
27         EXPIRATION_DATE=$(chage -l -i $USER | grep "Account expires" | awk -F:
28 '{print $2}' | xargs)
29
30         # Se a conta não tem data de expiração, pula para o próximo usuário
31         if [ "$EXPIRATION_DATE" == "never" ]; then
32             continue
33         fi
34
35         # Converte a data de expiração para segundos desde Epoch
36         if ! EXPIRATION_DATE_EPOCH=$(date -d "$EXPIRATION_DATE" +%s 2>/dev/null);
37 then
38             echo "Data de expiração inválida para o usuário $USER: $EXPIRATION_DATE"
39             continue
40         fi
41
42         # Calcula a diferença em dias entre a data atual e a data de expiração
43         DIFF_DAYS=$(( (CURRENT_DATE - EXPIRATION_DATE_EPOCH) / (60*60*24) ))
```

```

43         # Se a diferença em dias for maior que o número de dias fornecido, apaga o
usuário
44         if [ "$DIFF_DAYS" -gt "$DAYS" ]; then
45             echo "Apagando usuário $USER (expirado há $DIFF_DAYS dias)"
46             userdel -r -f $USER
47         fi
48     fi
49 done

```

Como Executar o Script

Salve o script em um arquivo chamado `delete_expired_users.sh`, dê permissão de execução e execute-o com o número de dias como argumento:

```

1  chmod +x delete_expired_users.sh
2  sudo ./delete_expired_users.sh numero_de_dias

```

Exemplo:

Para apagar todos os usuários do grupo `lamemo` cujas contas estejam expiradas há mais de 30 dias:

```

1  sudo ./delete_expired_users.sh 30

```

10.3 Removendo usuários de um grupo

Utilize o script `delete_group_users` para adicionar usuários novos:

```

1  #!/bin/bash
2
3  # Verifica se o script foi executado com privilégios de superusuário
4  if [ "$(id -u)" -ne 0 ]; then
5      echo "Este script precisa ser executado como root."
6      exit 1
7  fi
8
9  # Verifica se o nome do grupo foi fornecido como argumento
10 if [ -z "$1" ]; then
11     echo "Uso: $0 nome_do_grupo"
12     exit 1
13 fi
14
15 # Atribui o nome do grupo à variável GROUP
16 GROUP=$1
17
18 # Verifica se o grupo existe
19 if ! getent group "$GROUP" > /dev/null; then
20     echo "Grupo '$GROUP' não existe."
21     exit 1

```

```

22 fi
23
24 # Obtém a lista de todos os usuários do sistema
25 USERS=$(getent passwd | awk -F: '{print $1}')
26
27 # Remove cada usuário do grupo especificado
28 for USER in $USERS; do
29     # Verifica se o usuário pertence ao grupo especificado
30     if id -nG "$USER" | grep -qw "$GROUP"; then
31         echo "Apagando usuário $USER do grupo $GROUP"
32         userdel -r -f "$USER"
33     fi
34 done
35
36 echo "Todos os usuários do grupo '$GROUP' foram removidos."

```

Como Executar o Script

Salve o script em um arquivo chamado `delete_group_users.sh`, dê permissão de execução e execute-o com o nome do grupo como argumento:

```

1 chmod +x delete_group_users.sh
2 sudo ./delete_group_users.sh nome_do_grupo

```

Exemplo:

Para excluir todos os usuários do grupo `lamemo`:

```

1 sudo ./delete_group_users.sh alunos-hpc-2024-4

```

11. Configurando o envio de e-mails pelo Slurm

Para configurar o `sSMTP` para enviar emails através do servidor SMTP do Gmail usando uma senha de aplicativo (senha de app), siga os passos abaixo. O Gmail requer autenticação usando SSL/TLS para enviar emails, e a senha de aplicativo é necessária para aplicativos de terceiros como o `sSMTP`.

11.1 Passo a Passo para Configurar o sSMTP com Gmail

11.1.1. Gerar Senha de Aplicativo no Gmail

Para permitir que o `sSMTP` envie emails através do Gmail, você precisa gerar uma senha de aplicativo. Siga estes passos:

- **Passo 1:** Faça login na sua conta do Gmail.
- **Passo 2:** Acesse a página de Gerenciamento de Conta Google em <https://myaccount.google.com/apppasswords>

- **Passo 7:** Digite um nome para identificar o `ssmtp`, por exemplo, "ssmtp".
- **Passo 8:** Clique em "Criar".

Anote a senha de aplicativo gerada. Esta será a senha que você deve usar no campo `AuthPass` no arquivo `/etc/ssmtp/ssmtp.conf`.

11.1.2. Configuração inicial

1. Instale o pacote ssmtp:

```
1 | sudo apt-get install ssmtp
```

2. Configure o arquivo de configuração `/etc/ssmtp/ssmtp.conf` com suas credenciais de e-mail:

```
1 | root=seu_email@gmail.com
2 | mailhub=smtp.gmail.com:587
3 | hostname=seu_hostname
4 | AuthUser=seu_email@gmail.com
5 | AuthPass=sua_senha_do_gmail
6 | UseTLS=YES
7 | UseSTARTTLS=YES
```

Substitua `seu_email@gmail.com`, `sua_senha_do_gmail` e `seu_hostname` pelos valores apropriados.

3. Certifique-se de que o Gmail está configurado para permitir "Aplicativos menos seguros" ou "Acesso a app não seguro" nas configurações de segurança da sua conta do Google.

11.1.3. Configuração do script

Crie um novo arquivo de script, por exemplo, `/etc/slurm/slurm-email.sh`, e cole o seguinte conteúdo:

```
1 | #!/bin/bash
2 |
3 | body="$2"
4 | recipients="$3"
5 |
6 | # Extrair Job ID, Name e Status do corpo da mensagem
7 | job_id=$(echo "$body" | sed -n 's/.*Job_id=\([0-9]*\).*\/\1/p')
8 | name=$(echo "$body" | sed -n 's/.*Name=\([^\ ]*\).*\/\1/p')
9 | status=$(echo "$body" | sed -n 's/.*Name=[^\ ]* \([^\,]*\).*\/\1/p')
10 |
11 | # Formatar o assunto
12 | subject="Slurm Job ID: $job_id (Status: $status, Name: $name)"
13 |
14 | # Obter os caminhos dos arquivos de saída e erro do SLURM
15 | output_file="$(scontrol show job $job_id | grep -oP '(?<=StdOut=)\S+')"
16 | error_file="$(scontrol show job $job_id | grep -oP '(?<=StdErr=)\S+')"
17 |
18 | # Criar o corpo do email em HTML
19 | email_body="<html><body>"
20 | email_body+="<p>$body</p>"
```

```

21
22 # Tentar ler o arquivo de saída
23 output_content=$(sudo cat "$output_file" 2>/dev/null)
24 if [ -n "$output_content" ]; then
25     email_body+="

### Conteúdo do arquivo de saída ($output_file):</h3>" 26 email_body+=" ``` $output_content</pre>" 27 else 28 email_body+=" ```


```

Modifique a permissão do arquivo para permitir execução

```
1 | sudo chmod +x /etc/slurm/slurm-email.sh
```

Este script extrai informações relevantes do corpo da mensagem enviada pelo Slurm, como o ID do trabalho, nome e status. Em seguida, ele obtém os caminhos dos arquivos de saída e erro do Slurm, lê seu conteúdo (se existirem) e cria um corpo de e-mail em formato HTML contendo essas informações.

11.1.4. Atualizando o arquivo /etc/slurm/slurm.conf

1. Em todos os nós (`masternode` e `workernodes`) execute:

```
1 | echo 'MailProg=/etc/slurm/slurm-email.sh' | sudo tee -a /etc/slurm/slurm.conf
>/dev/null
```

2. Reinicie os serviços

No `masternode`:

```
1 | sudo systemctl restart slurmctld
```

No `workernode`:

```
1 | sudo systemctl restart slurmd
```

11.1.5. Considerações de segurança

Pressupõe que o usuário `slurm` tem permissão para executar o comando `sudo cat` sem senha para ler os arquivos de saída e erro do Slurm. Isso é feito adicionando uma linha ao arquivo `/etc/sudoers`:

1. Abra o arquivo `/etc/sudoers` para edição usando o comando `visudo`:

```
1 | sudo visudo
```

2. Adicione a seguinte linha no final do arquivo `/etc/sudoers`:

```
1 | slurm ALL=(root) NOPASSWD: /bin/cat /storage/home/*
```

Nota: Lembre-se de que é importante revisar cuidadosamente as permissões concedidas e garantir que elas sejam as mínimas necessárias para a funcionalidade desejada, seguindo os princípios de menor privilégio e necessidade de conhecer.

11.1.6. Exemplo de uso

A configuração de e-mail faz uso dos seguintes parâmetros: `--job-name`, `--output`, `--error`

1. Abra o arquivo `/etc/sudoers` para edição usando o comando `visudo`:

```
1 | #!/bin/bash
2 | #SBATCH --job-name=meu_job
3 | #SBATCH --output=meu_job.out
4 | #SBATCH --error=meu_job.err
5 | #SBATCH --partition=filal
6 | #SBATCH --nodes=1
7 | #SBATCH --ntasks=2
8 | #SBATCH --cpus-per-task=1
9 | #SBATCH --mem=1G
10 | #SBATCH --time=01:00:00
11 | #SBATCH --mail-type=ALL
12 | #SBATCH --mail-user=email@exemplo.com
13 |
14 | date
15 | echo "Job iniciado no nó: $(hostname)"
16 | echo "Usando $(nproc) núcleos de CPU"
17 | lsb_release -a
18 | echo "Job finalizado"
```

Considerações Finais

Esse documento apresenta uma configuração funcional e os comandos que devem permitir que os usuários utilizem o Slurm de maneira eficiente em seu ambiente HPC. Dependendo das necessidades específicas e das políticas do ambiente, você pode/deve ajustar e expandir essa configuração.