**Instalação e configurações adicionais KVM**

**Pop OS Nmcli**

***Criando uma interface bridge para Lab:***

**Adicionando a interface:**

Sudo nmcli connection add type bridge ifname lab1 autoconnect yes

**Conectando ela:**

sudo nmcli connection add type bridge-slave ifname eth1 master bridge-lab1 autoconnect yes

**Desabilitando a Conexão principal:**

nmcli connection down "Conexão cabeada 1"

**Habilitando a bridge-laboratorio:**

nmcli connection up "bridge-lab1"

**Reiniciando a rede:**

sudo systemctl restart network-online.target

***Desativando o netfilter para a ponte em Bridge:***

Para permitir que todo o tráfego seja encaminhado para a ponte e, portanto, para as máquinas virtuais conectadas a ela, precisamos desabilitar o netfilter. Isso é necessário, por exemplo, para que a resolução DNS funcione nas máquinas convidadas conectadas à ponte. Para fazer isso, podemos criar um arquivo com a extensão dentro do diretório, vamos chamá-lo de . Dentro dele escrevemos o seguinte conteúdo:.conf

Nano /etc/sysctl.d99-netfilter-bridge.conf

net.bridge.bridge-nf-call-ip6tables = 0

net.bridge.bridge-nf-call-iptables = 0

net.bridge.bridge-nf-call-arptables = 0

**Para carregar as configurações escritas no arquivo, garantimos que o módulo seja carregado**:br\_netfilter

$ sudo modprobe br\_netfilter

**Para carregar o módulo automaticamente na inicialização, vamos criar o arquivo: ele deve conter apenas o nome do próprio módulo:**

Nano /etc/modules-load.d/br\_netfilter.conf

br\_netfilter

**Uma vez que o módulo é carregado, para carregar as configurações que armazenamos no arquivo, podemos executar:** 99-netfilter-bridge.conf

$ sudo sysctl -p /etc/sysctl.d/99-netfilter-bridge.conf

Agora registraremos a nova interface para que seja visível durante a escolha de interfaces no virt-manager.

Este comando criará um arquivo xml chamado config-bridge.xml, contendo as informações necessárias para declarar uma interface no virsh e virt-manager.

cat <<EOF> config-bridge.xml

<network>

<name>laboratorio1</name>

<forward mode="bridge" />

<bridge name="lab1" />

</network>

EOF

sudo virsh net-define config-bridge.xml

sudo virsh net-start --network lab1

sudo virsh net-autostart lab1

**Para Firewall UFW podemos adicionar as seguintes etapas**:

# nano /etc/ufw/before.rules

Role até o final do arquivo e adicione as seguintes linhas – substituindo nn.nn.nn.nn pelo endereço IP da máquina virtual).

# Allow all traffic to nn.nn.nn.nn

-A FORWARD -d nn.nn.nn.nn -j ACCEPT

-A FORWARD -s nn.nn.nn.nn -j ACCEPT

# don't delete the 'COMMIT' line or these rules won't be processed

COMMIT

**Para que essas alterações entrem em vigor, você precisa reiniciar o ufw**

# ufw disable

# ufw enable

Systemctl restart ufw

**Esse passo abaixo se assegura que firewallD não bloqueie o tráfego ipv4 e a obtenção de endereços IPV4.**

Configurando o FirewallD para permitir o acesso à rede da máquina virtual em bridge

sudo firewall-cmd --permanent --direct --passthrough ipv4 -I FORWARD -i laboratorio1 -j ACCEPT

sudo firewall-cmd --permanent --direct --passthrough ipv4 -I FORWARD -o laboratorio1 -j ACCEPT

sudo firewall-cmd --reload

**Removendo a bridge, caso precise remover a bridge criada siga os seguintes procedimentos, substitua o laboratorio1 pelo sua interface bridger**

sudo virsh net-destroy --network lab1

sudo virsh net-undefine --network lab1

sudo virsh net-list --all

sudo nmcli c delete bridge-lab1