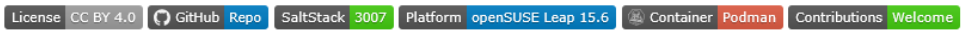
Guia de Configuração de Salt Master com Podman no openSUSE Leap



Salt Master (v3007) no Podman (openSUSE Leap) com SSH e Salt API

Este guia implementa um Salt Master com acesso SSH e Salt API no Podman, usando o openSUSE Leap 15.6 para garantir a estabilidade das dependências. A estrutura de diretórios e nomenclatura segue as melhores práticas do SaltStack.

1. Configuração Inicial e Estrutura de Arquivos

O projeto requer um diretório base, onde a imagem será construída, e os diretórios de persistência no host.

Comandos de Preparação do Host

a. Crie o diretório do projeto

```
mkdir salt-master-final
cd salt-master-final
```

b. Crie os diretórios de persistência no host (serão montados no contêiner)

```
mkdir -p /opt/salt-master_data/{keys,cache,files,pillar}
```

c. Crie os arquivos necessários para o build

```
touch Dockerfile
touch master_api.conf
touch run.sh
```

1.1. Arquivo: master_api.conf (Autenticação)

Este arquivo configura o Salt API na porta 8000 para usar a autenticação **auto** (chaves mestras), o que é mais estável em contêineres e evita a falha do módulo PAM.

```
# /etc/salt/master.d/master_api.conf
rest_cherrypy:
  port: 8000
  host: 0.0.0.0
  ssl_crt: /etc/pki/tls/certs/localhost.crt
  ssl_key: /etc/pki/tls/certs/localhost.key
external_auth:
  auto:
    root:
      - .*
      - '@runner'
      - '@wheel'
```

1.2. Arquivo: run.sh (Orquestração de Serviços)

Este script será o ENTRYPOINT (PID 1) do contêiner. Ele é configurado para iniciar o SSH Daemon e o Salt Master (que inicia a API internamente) em *background* e garantir que ambos os serviços permaneçam ativos.

```
#!/bin/bash

# Define a senha do root para acesso SSH.
echo "root:root" | chpasswd

# Geração de chaves SSH
ssh-keygen -A

# Inicia o serviço SSH
/usr/sbin/sshd

# 1. Inicia o Salt Master em background
/usr/bin/salt-master -l info &

# 2. Inicia o Salt API em background
/usr/bin/salt-api -l info &

# O comando 'wait' é NECESSÁRIO para manter o contêiner rodando, esperando pelos
processos em background.
wait -n
```

2. Dockerfile Final (openSUSE Leap 15.6)

Este Dockerfile é a versão final que corrige falhas de dependência (cherrypy, rpm-vercmp), garante o acesso SSH e estabelece permissões para o UID 1000.

```
Dockerfile
FROM opensuse/leap:15.6

# Variável de ambiente para o Shell
ENV SHELL /bin/bash

# 1. Criação Explícita de Usuário/Grupo
# Usamos UID/GID 1000 para evitar conflitos com grupos de sistema como 'users' (GID
100).
RUN groupadd -r -g 1000 salt && \
    useradd -r -u 1000 -g salt -m -s /bin/bash salt && \
    chsh -s /bin/bash root

# 2. Instalação de Pacotes, Chaves e Ajuste de Permissões
RUN zypper refresh && \
    # Instala pacotes base e PIP
    zypper install -y salt-master salt-api openssh vim curl python3-pip && \
    # CORREÇÃO CRÍTICA: Instala dependências Python que faltavam (cherrypy, rpm-vercmp)
via PIP
    pip3 install cherrypy rpm-vercmp && \
    # CRÍTICO: Permite login root por senha para o SSH, resolvendo o 'Permission denied'
do SSH
    sed -i 's/^#PermitRootLogin prohibit-password/PermitRootLogin yes/'
/etc/ssh/sshd_config && \
    # Cria todos os diretórios necessários (persistentes, logs e runtime)
    mkdir -p /srv/salt /srv/pillar /etc/salt/master.d /etc/salt/pki/master
/var/cache/salt/master /var/log/salt /var/run/salt /var/run/sshd /etc/pki/tls/certs && \
    # Geração de chaves SSH e SSL
    ssh-keygen -A && \
    openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
/etc/pki/tls/certs/localhost.key -out /etc/pki/tls/certs/localhost.crt -subj "/CN=salt-
master" && \
    # CRÍTICO: Ajusta a posse dos arquivos de chave SSL e diretórios para o UID 1000
(usuário 'salt')
    chown salt:salt /etc/pki/tls/certs/localhost.key /etc/pki/tls/certs/localhost.crt &&
\
    chown -R salt:salt /srv/salt /srv/pillar /etc/salt /var/cache/salt /var/log/salt
/var/run/salt && \
    # Limpeza de cache
    zypper clean -a

# 3. Copia a configuração da API e o script de inicialização
COPY master_api.conf /etc/salt/master.d/master_api.conf
COPY run.sh /usr/local/bin/run.sh

RUN chmod +x /usr/local/bin/run.sh
```

```
# Configura as portas (4505/4506 Salt, 8000 API, 22 SSH)
EXPOSE 4505
EXPOSE 4506
EXPOSE 8000
EXPOSE 22

USER root
ENTRYPOINT ["/usr/local/bin/run.sh"]
CMD [""]
```

3. Execução no Host

Este é o comando que inicia o Salt Master com todas as configurações, volumes e portas mapeadas.

Ações Finais no Host

1. **Ajuste de Permissão Crítico:**

   Garante que o diretório de persistência do host pertença ao UID 1000

```
sudo chown -R 1000:1000 /opt/salt-master_data
```

2. **Construção da Imagem:**

```
podman build -t salt-master-suse:3007.0 .
```

3. **Comando de Execução:**

```
podman run -d \
  --name salt-master \
  -p 10022:22/tcp \
  -p 4505:4505/tcp \
  -p 4506:4506/tcp \
  -p 8000:8000/tcp \
  -v /opt/salt-master_data/keys:/etc/salt/pki/master:Z \
  -v /opt/salt-master_data/cache:/var/cache/salt/master:Z \
  -v /opt/salt-master_data/files:/srv/salt:Z \
  -v /opt/salt-master_data/pillar:/srv/pillar:Z \
  --restart=always \
  salt-master-suse:3007.0
```

4. Estrutura de Automação (Salt Formulas)

A automação no SaltStack é baseada em **States** (Estados) e **Formulas**, armazenados no **Fileserver** (mapeado para /opt/salt-master_data/files).

Componentes de Automação

Diretório no Host (Fileserver)	Conteúdo	Propósito
/opt/salt-master_data/files	Arquivos .sls e o top.sls.	**State Files** (Definição da Configuração).

Diretório no Host (Fileserver)	Conteúdo	Propósito
/opt/salt-master_data/pillar	Arquivos .sls de Pillar.	**Pillar Data** (Dados sensíveis ou específicos do Minion).

Estrutura de Fórmulas (Exemplo NGINX)

A melhor prática é criar uma estrutura de diretórios para cada serviço (uma "Formula").

Arquivo/Diretório	Localização	Conteúdo e Justificativa
Formula Directory	/opt/salt-master_data/files/nginx/	**Melhor Prática:** Agrupa todos os states do NGINX.
State File	/opt/salt-master_data/files/nginx/init.sls	**Convenção:** O arquivo principal de um diretório, chamado usando o nome do diretório.

Conteúdo do init.sls:

YAML

```
nginx_package:
  pkg.installed:
    - name: nginx

nginx_service:
  service.running:
    - name: nginx
    - enable: True
    - require:
      - pkg: nginx_package
```

Comando de Execução do State:

```
ssh root@<IP_VM> -p 10022
salt '*' state.apply nginx  # Chama o arquivo nginx/init.sls
```