

Управление виртуальными машинами с помощью libvirt

libvirt — это мощная коллекция инструментов для управления виртуализацией. Она предоставляет унифицированный API для взаимодействия с различными гипервизорами, такими как KVM, Xen и другие. В этой документации мы рассмотрим введение в libvirt и его архитектуру, процесс установки и настройки libvirt, основные команды для управления виртуальными машинами, а также настройку сети и хранения через libvirt на НАЙС ОС.

Введение в libvirt и его архитектуру

libvirt — это библиотека и набор инструментов для управления виртуализацией. Она предоставляет API для управления виртуальными машинами, а также инструмент командной строки `virsh` для взаимодействия с гипервизорами.

Основные компоненты libvirt

- **libvirtd**: демон, отвечающий за управление виртуальными машинами и взаимодействие с гипервизорами.
- **libvirt API**: программный интерфейс, который позволяет разработчикам создавать приложения для управления виртуальными машинами.
- **virsh**: утилита командной строки для управления виртуальными машинами и гипервизорами через libvirt.
- **virt-manager**: графический интерфейс для управления виртуальными машинами.

Архитектура libvirt

libvirt имеет модульную архитектуру, которая включает в себя:

- Поддержку различных гипервизоров через драйверы (KVM, Xen, VMware, VirtualBox и др.).
- Поддержку управления сетями и хранилищами.
- Механизмы безопасности и изоляции.
- Расширяемость через плагины и API.

Установка и настройка libvirt

Для установки libvirt на НАЙС ОС используйте пакетный менеджер `tdnf`. После установки необходимо настроить и запустить службы libvirtd.

Установка libvirt на НАЙС ОС

Выполните следующие команды для установки libvirt:

```
sudo tdnf install libvirt virt-install qemu-kvm bridge-utils
sudo systemctl enable --now libvirtd
```

Настройка libvirt

После установки необходимо выполнить некоторые настройки, чтобы libvirt мог полноценно работать с виртуальными машинами.

Проверка состояния службы libvirtd

Убедитесь, что служба libvirtd запущена:

```
sudo systemctl status libvirtd
```

Настройка пользователя

Добавьте своего пользователя в группу libvirt для управления виртуальными машинами без необходимости использования прав суперпользователя:

```
sudo usermod -aG libvirt $(whoami)  
newgrp libvirt
```

Основные команды для управления виртуальными машинами

Основные команды для управления виртуальными машинами с помощью virsh включают создание, запуск, остановку и удаление виртуальных машин.

Создание новой виртуальной машины

Используйте команду virt-install для создания новой виртуальной машины:

```
virt-install --name myvm --ram 2048 --vcpus 2 --disk size=20 --os-variant  
ubuntu20.04 --network bridge=br0 --cdrom /path/to/ubuntu.iso
```

Запуск виртуальной машины

Для запуска виртуальной машины используйте команду virsh start:

```
virsh start myvm
```

Остановка виртуальной машины

Для остановки виртуальной машины используйте команду virsh shutdown:

```
virsh shutdown myvm
```

Удаление виртуальной машины

Для удаления виртуальной машины используйте команды `virsh undefine` и `virsh vol-delete`:

```
virsh undefine myvm
virsh vol-delete --pool default myvm.img
```

Настройка сети и хранения через libvirt

Настройка сети и хранения для виртуальных машин является ключевым аспектом управления виртуализацией. libvirt предоставляет гибкие инструменты для настройки сетевых мостов и хранилищ.

Настройка сети

Для настройки сети можно использовать сетевые мосты, что позволяет виртуальным машинам использовать физическую сеть хоста.

Создание сетевого моста

Создайте сетевой мост и настройте его для использования с виртуальными машинами:

```
sudo nmcli connection add type bridge con-name br0 ifname br0
sudo nmcli connection modify br0 ipv4.addresses "192.168.1.100/24" ipv4.method
manual
sudo nmcli connection add type ethernet slave-type bridge con-name br0-port ifname
eth0 master br0
sudo nmcli connection up br0
```

Настройка виртуальных машин для использования сетевого моста

При создании новой виртуальной машины укажите использование сетевого моста:

```
virt-install --name myvm --ram 2048 --vcpus 2 --disk size=20 --os-variant
ubuntu20.04 --network bridge=br0 --cdrom /path/to/ubuntu.iso
```

Настройка хранения

libvirt поддерживает различные типы хранилищ, включая директории, LVM и NFS. Рассмотрим создание и использование хранилища на основе директории.

Создание пула хранения

Создайте новый пул хранения, используя директорию на хосте:

```
sudo mkdir /var/lib/libvirt/images
```

```
virsh pool-define-as default dir - - - - "/var/lib/libvirt/images"  
virsh pool-build default  
virsh pool-start default  
virsh pool-autostart default
```

Создание тома хранения

Создайте новый том хранения в пуле:

```
virsh vol-create-as default myvm.img 20G
```

Использование тома хранения при создании виртуальной машины

При создании новой виртуальной машины укажите использование созданного тома:

```
virt-install --name myvm --ram 2048 --vcpus 2 --disk  
path=/var/lib/libvirt/images/myvm.img --os-variant ubuntu20.04 --network  
bridge=br0 --cdrom /path/to/ubuntu.iso
```

Мониторинг и управление виртуальными машинами

После настройки виртуальных машин важно обеспечить их мониторинг и управление для поддержания стабильной работы.

Мониторинг состояния виртуальных машин

Используйте команду `virsh list` для отображения списка работающих виртуальных машин:

```
virsh list --all
```

Проверка использования ресурсов

Используйте команду `virt-top` для мониторинга использования ресурсов виртуальными машинами:

```
virt-top
```

Управление снапшотами

Создание и управление снапшотами позволяет сохранять состояние виртуальной машины для последующего восстановления.

Создание снапшота

Для создания снапшота используйте команду `virsh snapshot-create-as`:

```
virsh snapshot-create-as myvm myvm-snapshot
```

Восстановление из снапшота

Для восстановления виртуальной машины из снапшота используйте команду `virsh snapshot-revert`:

```
virsh snapshot-revert myvm myvm-snapshot
```

Расширенные настройки libvirt

libvirt предоставляет множество расширенных настроек для улучшения производительности и безопасности виртуальных машин.

Настройка привязки CPU (CPU Pinning)

Привязка виртуальных процессоров (vCPU) к физическим ядрам может улучшить производительность, особенно на многопроцессорных системах.

Используйте команду `virsh vcpupin` для привязки vCPU к конкретным ядрам:

```
virsh vcpupin myvm 0 2  
virsh vcpupin myvm 1 3
```

Настройка больших страниц памяти (HugePages)

Использование больших страниц памяти может уменьшить накладные расходы на управление памятью и улучшить производительность.

Настройте использование больших страниц памяти в конфигурации виртуальной машины:

```
...  
...  
...
```

Настройка политики ввода-вывода (I/O Threads)

Создание отдельных потоков ввода-вывода для дисков виртуальных машин может улучшить производительность.

Добавьте потоки ввода-вывода в конфигурацию виртуальной машины:

```
...  
2  
...
```

Управление виртуальными машинами с помощью libvirt предоставляет мощные инструменты для развертывания, мониторинга и настройки виртуализации. Правильная настройка сети и хранения, использование расширенных возможностей libvirt помогут вам оптимизировать работу виртуальных машин и обеспечить высокую производительность и стабильность вашей виртуальной инфраструктуры.