МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение   
высшего образования

«КРЫМСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ им. В. И. ВЕРНАДСКОГО»

ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ

Кафедра компьютерной инженерии и моделирования

**ОТЧЕТ ПО ПРАКТИЧЕСКОМУ ЗАДАНИЮ №4**

**«ВИРТУАЛИЗАЦИЯ.ЭМУЛЯЦИЯ.ГИПЕРВИЗОРЫ»**

Практическая работа

по дисциплине «Программная инженерия»

студента 1 курса группы ПИ-б-о-241(1)

Коробка Илья Леонидович

09.03.04 «Программная инженерия»

Симферополь, 2025

## **Теоретическая база**

### Основные понятия

**Виртуализация** - технология создания виртуальных (а не физических) версий компьютерных ресурсов: операционных систем, серверов, устройств хранения данных или сетевых ресурсов.

**Эмуляция** - полная имитация поведения одной системы с помощью другой системы. Эмулятор переводит каждую инструкцию гостевой системы в инструкции хост-системы.

**Гипервизор** - программа или аппаратно-программный комплекс, обеспечивающий одновременное выполнение нескольких операционных систем на одном физическом компьютере.

### **Типы гипервизоров:**

1. **Тип 1 (Bare Metal)**: работают непосредственно на аппаратном обеспечении

* Примеры: VMware vSphere/ESXi, Microsoft Hyper-V, Xen

1. **Тип 2 (Hosted)**: работают поверх операционной системы

* Примеры: VMware Workstation, VirtualBox, QEMU

### QEMU и KVM

**QEMU** (Quick Emulator) - гипервизор типа 2, способный работать в двух режимах:

* **Эмуляция**: полная программная эмуляция процессора и устройств
* **Виртуализация**: использование аппаратных возможностей процессора (с KVM)

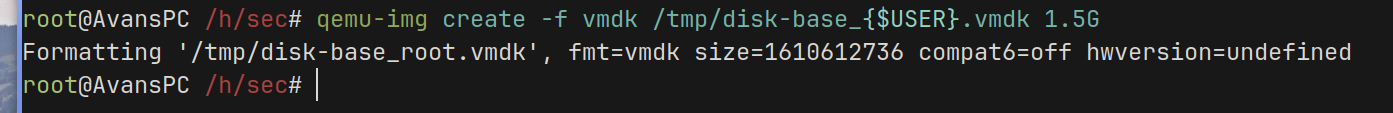
**KVM** (Kernel-based Virtual Machine) - модуль ядра Linux, превращающий Linux в гипервизор типа 1. Использует аппаратные расширения виртуализации Intel VT-x или AMD-V.

## **Пошаговое выполнение лабораторной работы**

### 1. Работа с qemu-img

#### 1.1 Создание образа диска в формате VMDK

qemu-img create -f vmdk /tmp/disk-base\_$USER.vmdk 1.5G



**Разбор команды:**

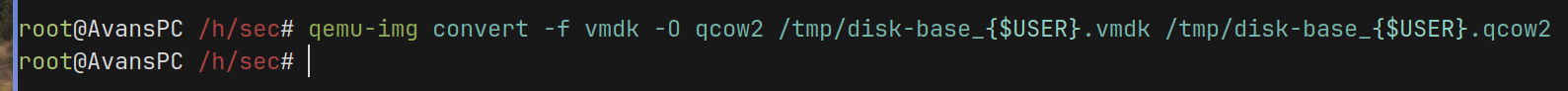
* qemu-img create - команда создания образа диска
* -f vmdk - формат образа (VMware Virtual Disk)
* /tmp/disk-base\_$USER.vmdk - путь и имя файла
* 1.5G - размер диска

**Форматы образов:**

* **VMDK**: формат VMware, совместим с VMware продуктами
* **QCOW2**: формат QEMU, поддерживает снимки, сжатие, шифрование
* **VDI**: формат VirtualBox
* **RAW**: несжатый образ, максимальная производительность

#### 1.2 Конвертация формата

qemu-img convert -f vmdk -O qcow2 /tmp/disk-base\_$USER.vmdk /tmp/disk-base\_$USER.qcow2

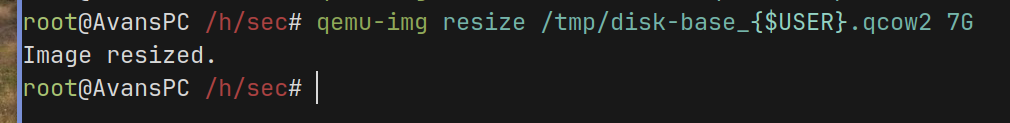


**Разбор команды:**

* convert - команда конвертации
* -f vmdk - исходный формат
* -O qcow2 - целевой формат
* Исходный и целевой файлы

#### 1.3 Изменение размера образа

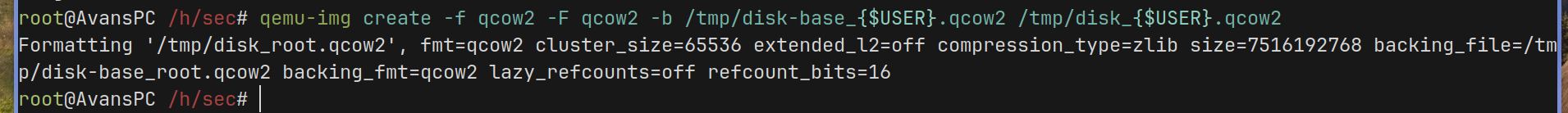
qemu-img resize /tmp/disk-base\_$USER.qcow2 7G



**Важно:** Это изменяет только размер образа, не файловую систему внутри!

#### 1.4 Создание дочернего образа

qemu-img create -f qcow2 -b /tmp/disk-base\_$USER.qcow2 /tmp/disk\_$USER.qcow2



**Разбор команды:**

* -b - backing file (базовый образ)
* Дочерний образ хранит только изменения относительно базового

**Преимущества дочерних образов:**

* Экономия места на диске
* Быстрое создание множества ВМ с одной базой
* Возможность отката к базовому состоянию

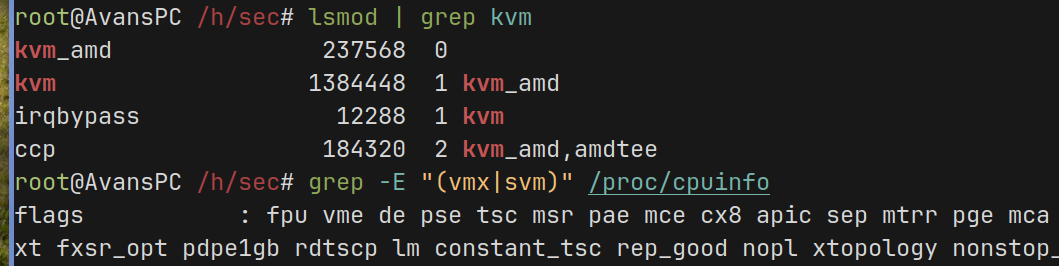
### 2. Проверка поддержки KVM

# Проверка модуля KVM

lsmod | grep kvm

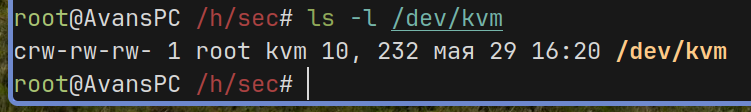
# Проверка поддержки аппаратной виртуализации

grep -E "(vmx|svm)" /proc/cpuinfo



# Проверка устройства KVM

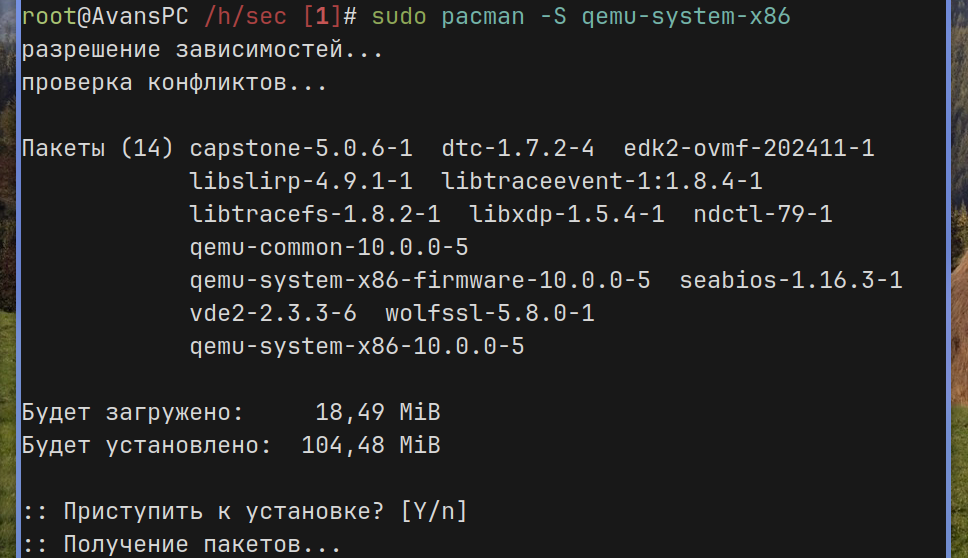
ls -l /dev/kvm



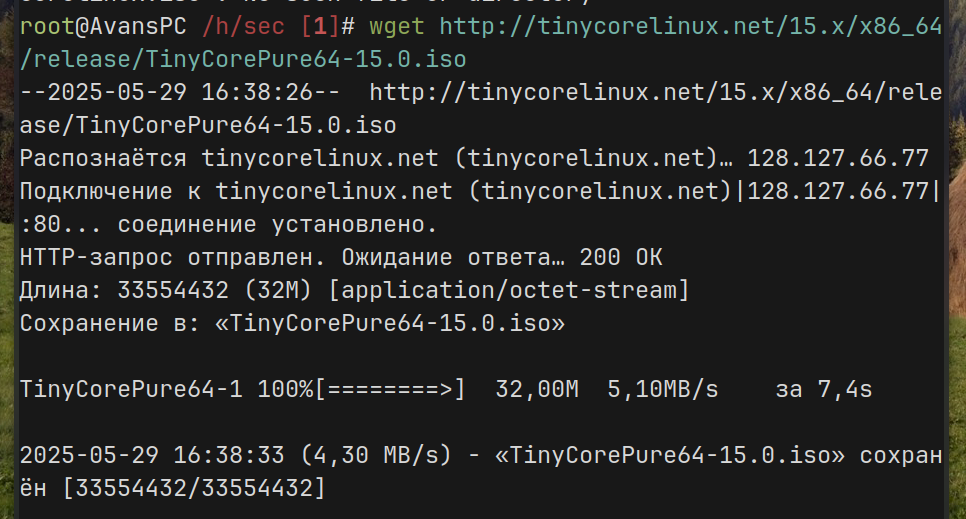
**Intel VT-x**: флаг vmx в /proc/cpuinfo **AMD-V**: флаг svm в /proc/cpuinfo

### 3. Запуск виртуальной машины

Для начала мне пришлось скачать qemu-system-x86 - программа для запуска гостевых ОС



А после и образ системы



qemu-system-x86\_64 \

-smp 1 \

-m 1.5G \

-vga std \

-hda /tmp/disk\_$USER.qcow2 \

-cdrom tinycorelinux.iso \

-netdev user,id=net0,hostfwd=tcp::8080-:80 \

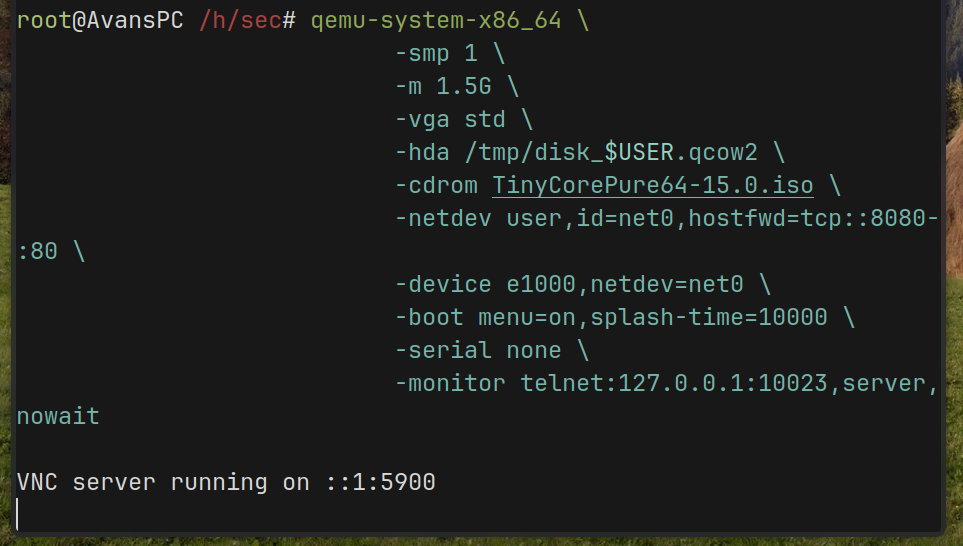
-device virtio-net-pci,netdev=net0 \

-boot menu=on,splash-time=10000 \

-enable-kvm \

-serial none \

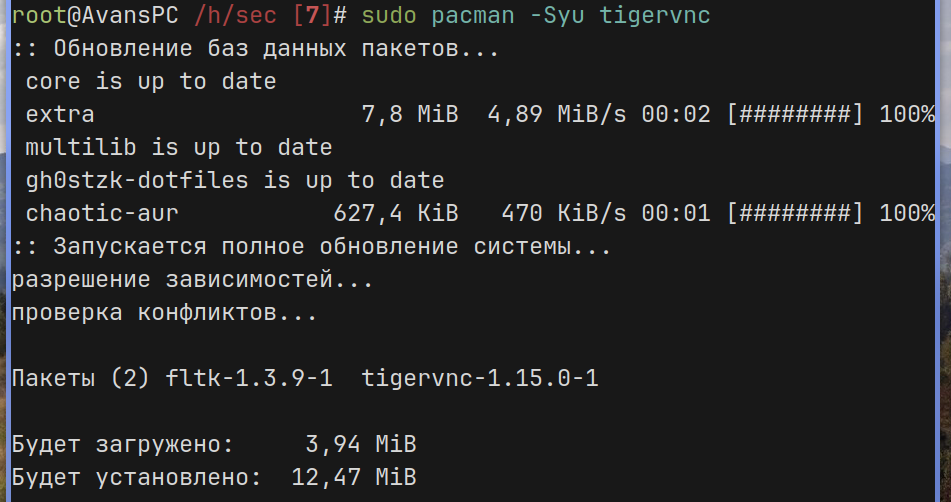
-monitor telnet:127.0.0.1:10023,server,nowait



**Детальный разбор параметров:**

* -smp 1: количество CPU (Symmetric MultiProcessing)
* -m 1.5G: объем оперативной памяти
* -vga std: тип видеокарты (std, cirrus, vmware, qxl, none)
* -hda: первый жесткий диск (IDE)
* -cdrom: CD-ROM образ
* -netdev user,id=net0,hostfwd=tcp::8080-:80:
* user - режим пользовательской сети (NAT)
* id=net0 - идентификатор сетевого устройства
* hostfwd=tcp::8080-:80 - проброс портов
* -device virtio-net-pci,netdev=net0: сетевая карта AMD
* -boot menu=on,splash-time=10000:
* menu=on - показать меню загрузки
* splash-time=10000 - таймаут в миллисекундах
* -enable-kvm: использовать KVM (аппаратная виртуализация)
* -serial none: отключить последовательный порт
* -monitor telnet:127.0.0.1:10023,server,nowait:
* монитор QEMU через telnet
* server - QEMU слушает подключения
* nowait - не ждать подключения для старта

Для подключения к ВМ пришлось скачать дополнительный софт



### 4. Установка пакетов в TinyCore Linux

tce-load -wi openssh

**TinyCore Linux** - минималистичная Linux-система:

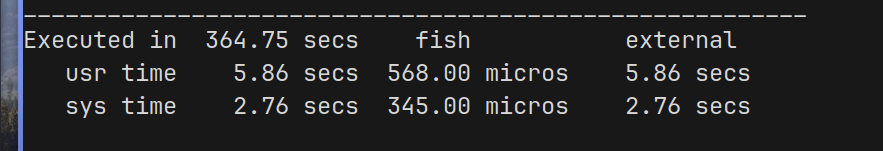
* tce-load - менеджер пакетов TinyCore
* -w - загрузить из интернета
* -i - установить пакет

### 5. Измерение производительности

#### Время загрузки системы

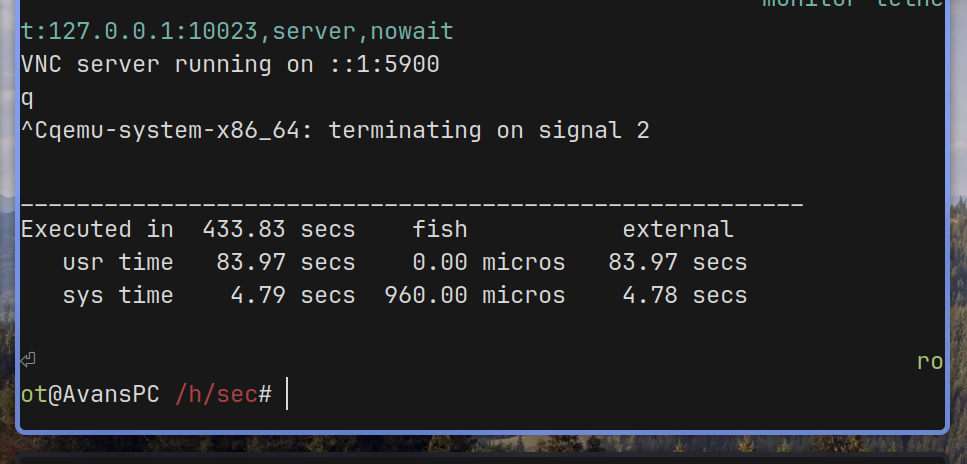
**С KVM (виртуализация):**

time qemu-system-x86\_64 -enable-kvm [остальные параметры]



**Без KVM (эмуляция):**

time qemu-system-x86\_64 [остальные параметры без -enable-kvm]



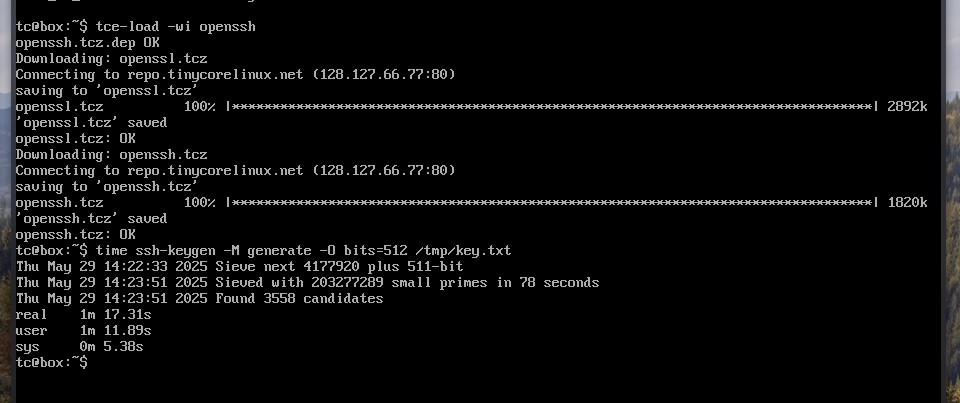
#### Время выполнения команды

В виртуальной машине:

time ssh-keygen -M generate -O bits=512 /tmp/key.txt

**Разбор команды ssh-keygen:**

* -M generate - генерировать модули для DH (Diffie-Hellman)
* -O bits=512 - размер ключа в битах
* /tmp/key.txt - выходной файл



### 6. Работа с монитором QEMU

#### Подключение к монитору

telnet 127.0.0.1 10023

**Отключение:** Ctrl+] затем Ctrl+d

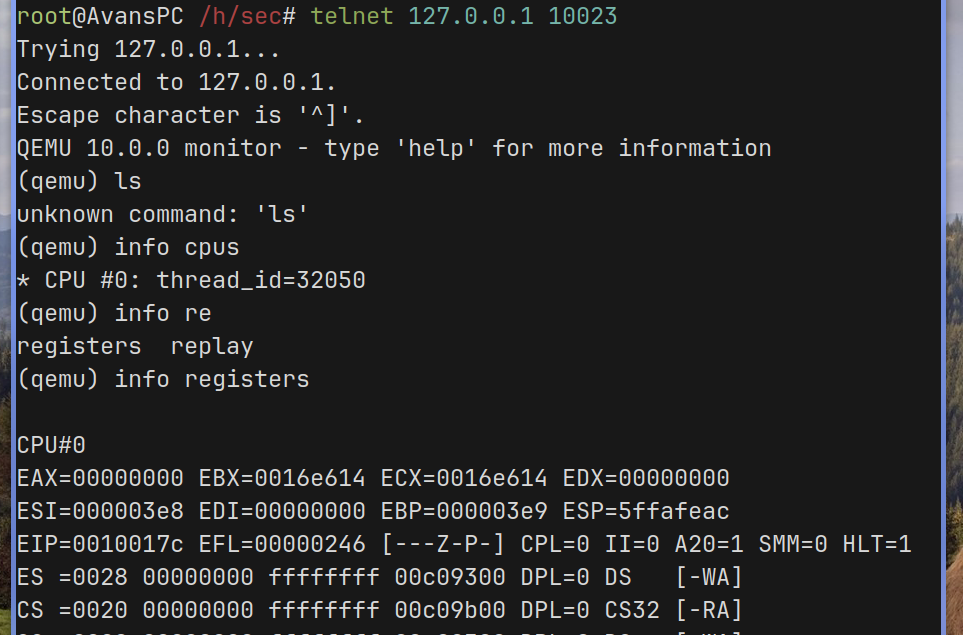
#### Основные команды монитора

**Информация о процессорах:**

info cpus

**Информация о регистрах:**

info registers

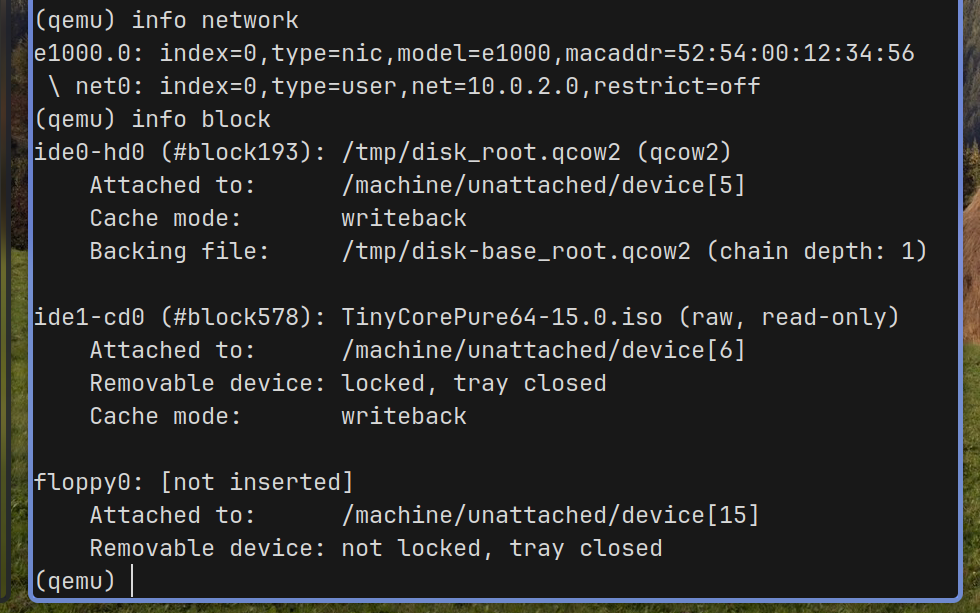
****

**Информация о сети:**

info network

**Информация о блочных устройствах:**

info block

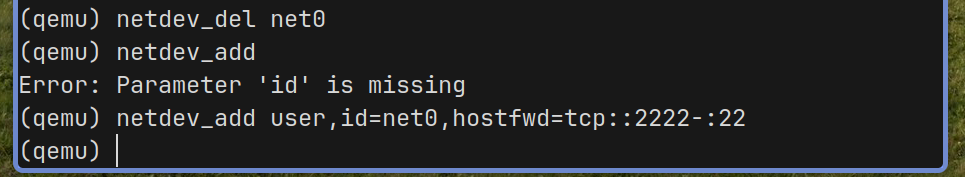


**Управление пробросом портов:**

# Удалить существующий проброс

netdev\_del net0

netdev\_add user,id=net0,hostfwd=tcp::2222-:22



**Управление снимками:**

# Сохранить состояние

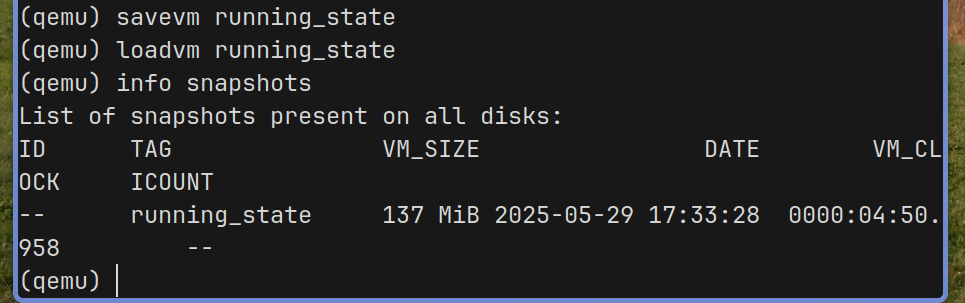
savevm running\_state

# Загрузить состояние

loadvm running\_state

# Список снимков

info snapshots



**Управление ВМ:**

# Перезагрузка (reset)

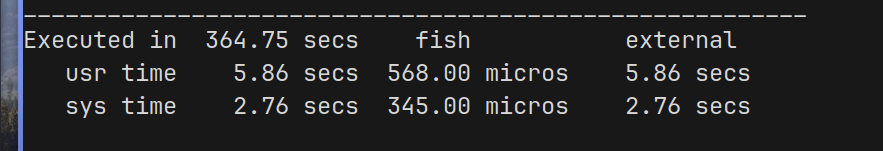
system\_reset

# Выключение

quit

# или

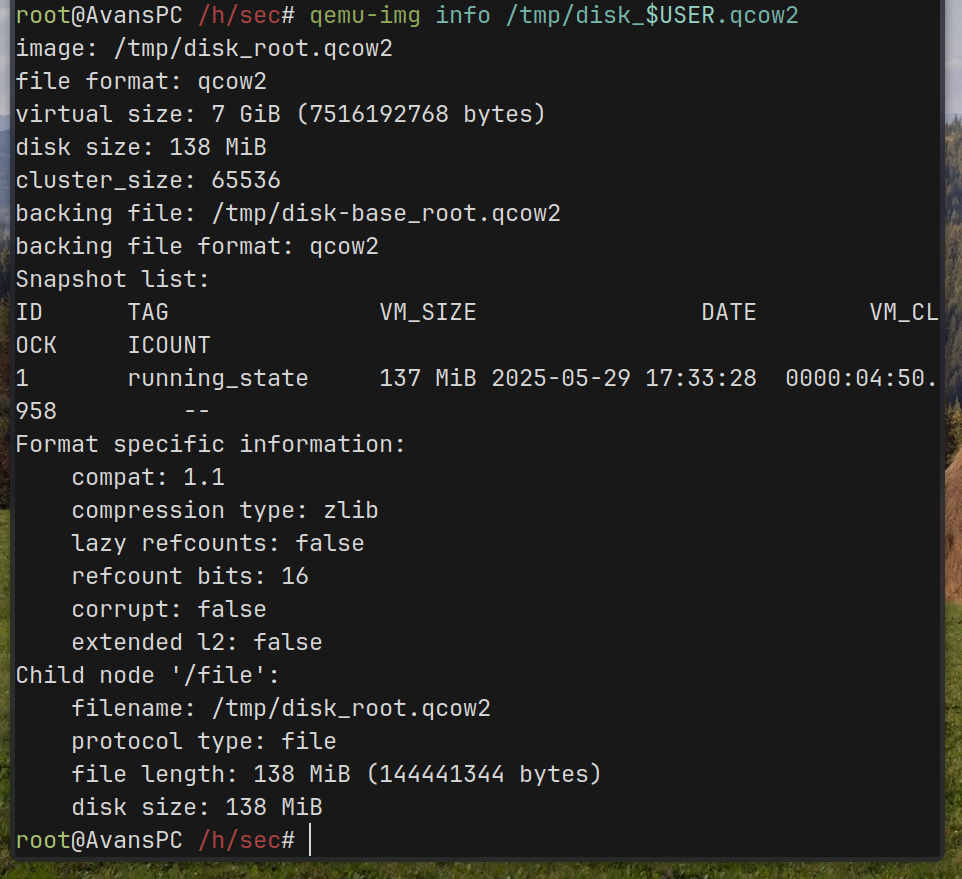
system\_powerdown



### 7. Работа с образами и снимками

#### Информация об образе

qemu-img info /tmp/disk\_$USER.qcow2

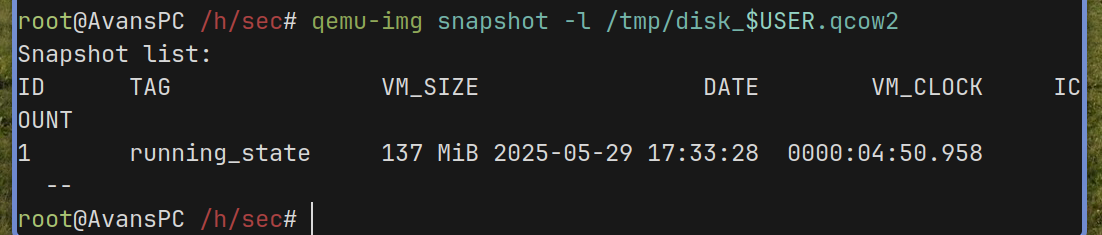


Покажет:

* Виртуальный размер
* Реальный размер на диске
* Формат
* Список снимков

#### Просмотр снимков

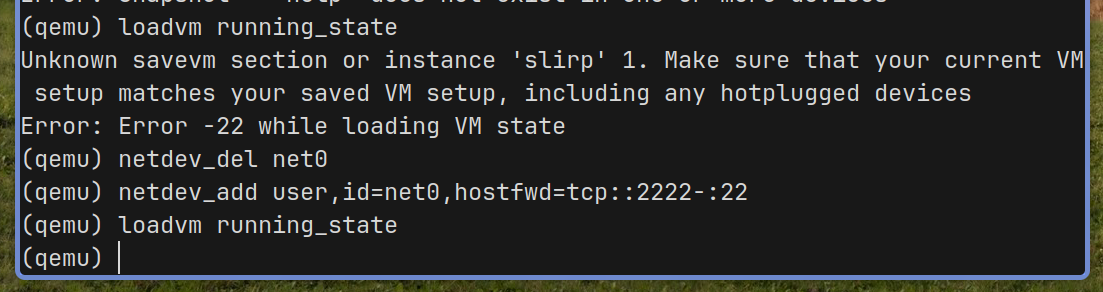
qemu-img snapshot -l /tmp/disk\_$USER.qcow2



#### Восстановление из снимка

В мониторе QEMU:

loadvm running\_state



## **Важные концепции**

### Различия между эмуляцией и виртуализацией

**Эмуляция:**

* Полная программная имитация оборудования
* Медленнее (10-100x замедление)
* Не требует аппаратной поддержки
* Может эмулировать другие архитектуры

**Виртуализация:**

* Использует аппаратные расширения процессора
* Близкая к нативной производительность (5-15% overhead)
* Требует VT-x/AMD-V
* Только та же архитектура

### Copy-on-Write в QCOW2

* Базовый образ остается неизменным
* Дочерний образ хранит только изменения
* При чтении: сначала дочерний, потом базовый
* При записи: создается копия блока в дочернем образе

### Типы сетей в QEMU

1. **User mode (SLIRP)**: NAT, проброс портов
2. **TAP**: мостовое соединение
3. **Socket**: соединение между ВМ
4. **VDE**: Virtual Distributed Ethernet
5. **В чем разница между QEMU и KVM?**

* QEMU - эмулятор/гипервизор, KVM - модуль ядра для аппаратной виртуализации

1. **Почему дочерние образы экономят место?**

* Используют Copy-on-Write, хранят только изменения

1. **Что происходит при savevm/loadvm?**

* Сохраняется полное состояние ВМ: память, регистры, устройства

1. **Зачем нужен проброс портов?**

* В режиме NAT ВМ недоступна извне, проброс открывает доступ к службам

1. **Что такое монитор QEMU?**

* Интерфейс управления ВМ во время работы

# Просмотр справки

man qemu-system-x86\_64

man qemu-img

qemu-img --help

qemu-system-x86\_64 -help

# Список поддерживаемых форматов

qemu-img --help | grep "Supported formats"

# Список поддерживаемых устройств

qemu-system-x86\_64 -device help