

Виртуализация: Типы виртуализаций KVM, QEMU



Александр
Зубарев



Александр Зубарев

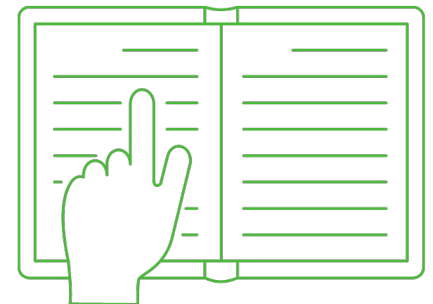
Председатель цикловой комиссии “Информационной
безопасности инфокоммуникационных систем”

АКТ (ф) СПбГУТ

Предисловие

На этом занятии мы:

- поговорим о типах виртуализации;
- рассмотрим, какие системы виртуализации бывают;
- узнаем, как работают виртуальные машины и распределяются ресурсы.





План занятия

1. [Введение](#)
2. [Типы виртуализации](#)
3. [История создания](#)
4. [Технологии аппаратной виртуализации](#)
5. [KVM](#)
6. [libvirt](#)
7. [XEN](#)
8. [OEMU](#)
9. [GNS 3](#)
10. [Итоги](#)
11. [Домашнее задание](#)



Введение

Термины виртуализации

Виртуализация — это создание изолированных окружений в рамках одного физического устройства.

Каждое окружение выглядит, как отдельный компьютер со своими характеристиками (доступная память, процессор и тп.)

Такое окружение называют набором **логических ресурсов** или **виртуальной машиной**.

ОС, внутри которой стартует другая ОС, называется **хост-системой, (host)**. А ОС, которая работает в виртуальном окружении — **гостевой (guest)**.



Типы виртуализации

Типы виртуализации

Виртуализация бывает:

- Аппаратная
- Программная
- Контейнерная (будет рассмотрена в следующих лекциях)
- Хостинговая (рассматривалась на первой лекции)



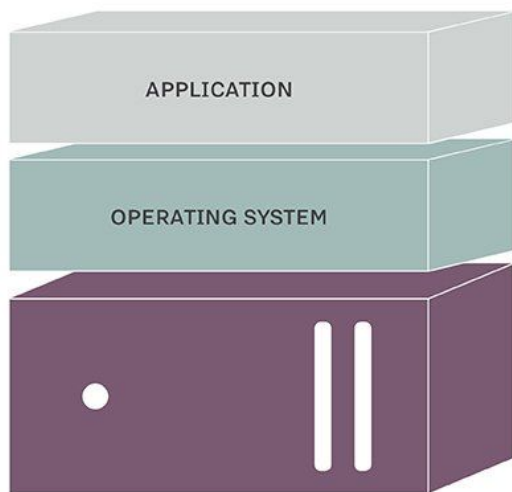
Типы виртуализации

Аппаратная виртуализация работает благодаря поддержке со стороны железа: процессора. В отличие от программной виртуализации, гостевые ОС управляются гипервизором напрямую, без участия хостовой ОС.

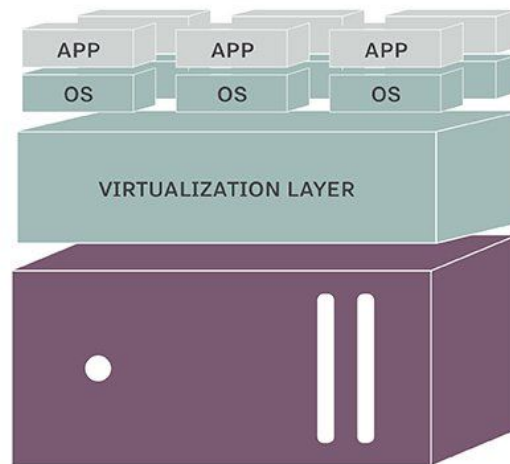
- + Работает без посредников, быстро, весело, надежно.
- Привязана к платформе: нет технологии — нет виртуализации.

Типы виртуализации

TRADITIONAL AND VIRTUAL ARCHITECTURE



TRADITIONAL ARCHITECTURE



VIRTUAL ARCHITECTURE



Типы виртуализации

Программная виртуализация эмулирует все железо от процессора до сетевого адаптера (если он нужен). В отличие от аппаратной виртуализации, не важно, какое у вас аппаратное обеспечение, будет работать на любом железе.

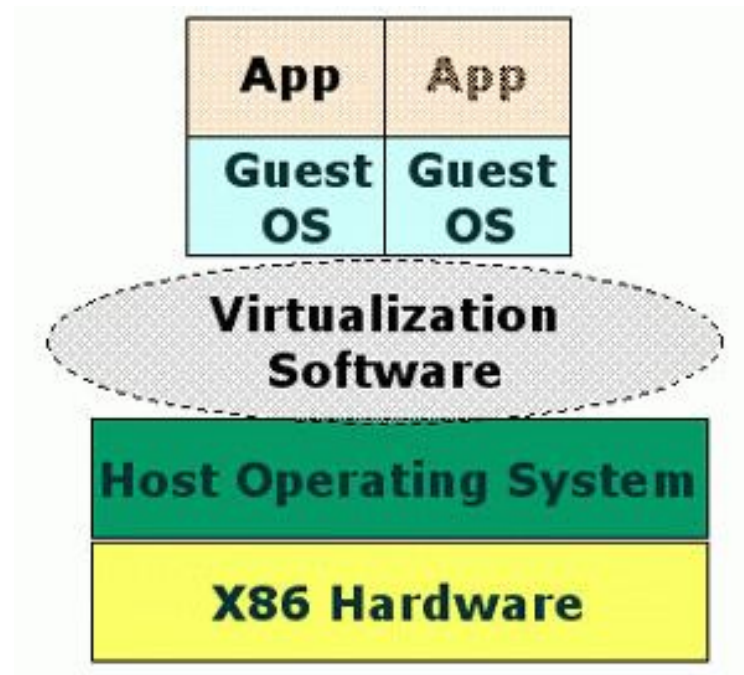
- + Хост система и гостевая система не зависимы.
- Медленная, так как надо эмулировать все, что работает быстро.

Типы виртуализации

Программную виртуализацию можно использовать для игр, но медленно.



источник



источник



История создания

История создания

Первая аппаратная виртуализация Intel была воплощена в 386-х процессорах и носила название V86 mode.

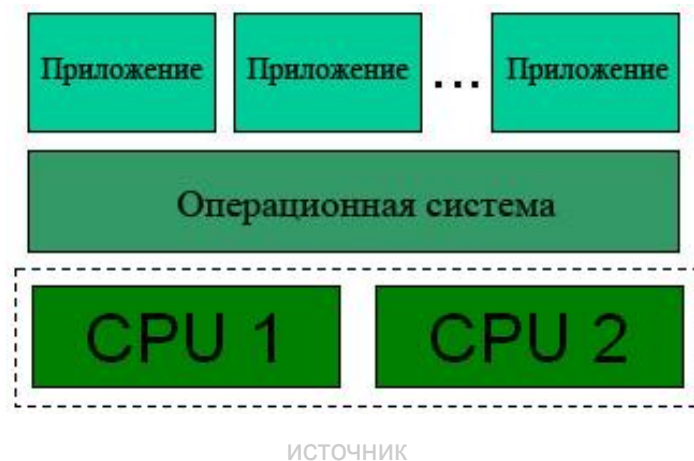
Этот режим работы 8086-го процессора позволял запускать параллельно несколько DOS-приложений.



ИСТОЧНИК

История создания

Второе, что было реализовано — многозадачность. Она является первым уровнем абстракции приложений. Каждое приложение распределяет ресурсы физического процессора в режиме разделения исполнения кода по времени.



История создания

HyperThreading представляет собой аппаратную технологию виртуализации: при ее использовании происходит симуляция двух виртуальных процессоров в рамках одного физического с помощью техники Symmetric Multi Processing (SMP).






История создания

Процессор с поддержкой виртуализации может работать в двух режимах **root operation** и **non-root operation**.

В режиме **root operation** работает специальное программное обеспечение, являющееся «легковесной» прослойкой между гостевыми операционными системами и оборудованием — монитор виртуальных машин (Virtual Machine Monitor, VMM), носящий также название гипервизор (hypervisor).



Технологии аппаратной виртуализации

Технологии аппаратной виртуализации

Производители и технологии виртуализации

Intel — VT-x VT-d VMDQ



ИСТОЧНИК

AMD — AMD-V



ИСТОЧНИК

ARM Limited — EL2



ИСТОЧНИК

Примечание: Для работы аппаратной виртуализации в программной процессор должен иметь данные инструкции



Intel vs AMD

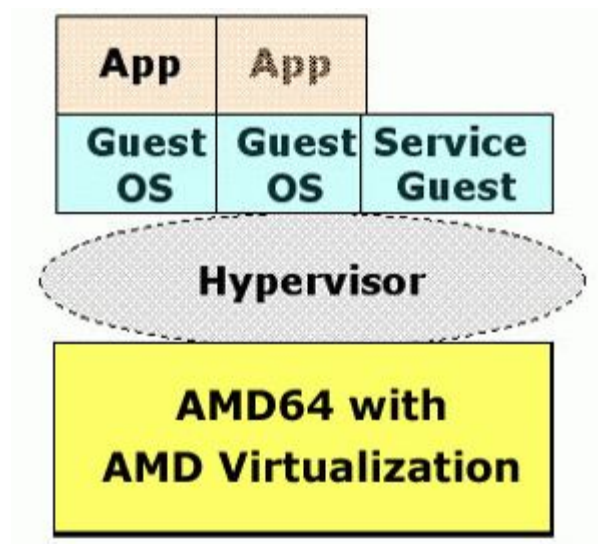
Ведущие производители процессоров для серверных и настольных платформ — компании Intel и AMD разработали техники аппаратной виртуализации, для их использования в платформах виртуализации.

И в 2005 году представили свои инструкции:

- VT-x
- AMD-V

AMD

Аппаратная виртуализация — это логическое развитие архитектуры AMD Direct Connect, реализующая технологию виртуализации в кремнии. Эта технология дает больше возможностей производителям программного обеспечения, позволяя не беспокоиться о программной эмуляции виртуализации на процессоре.

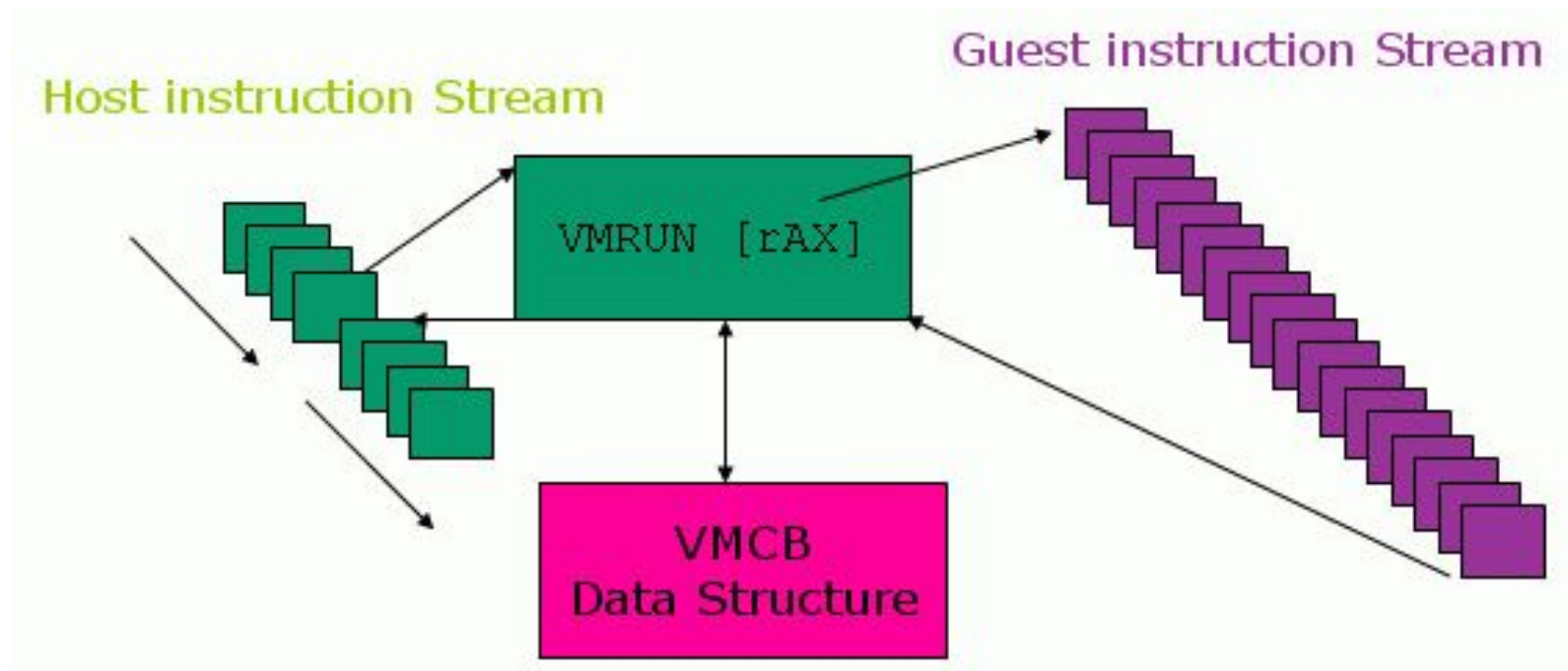


AMD

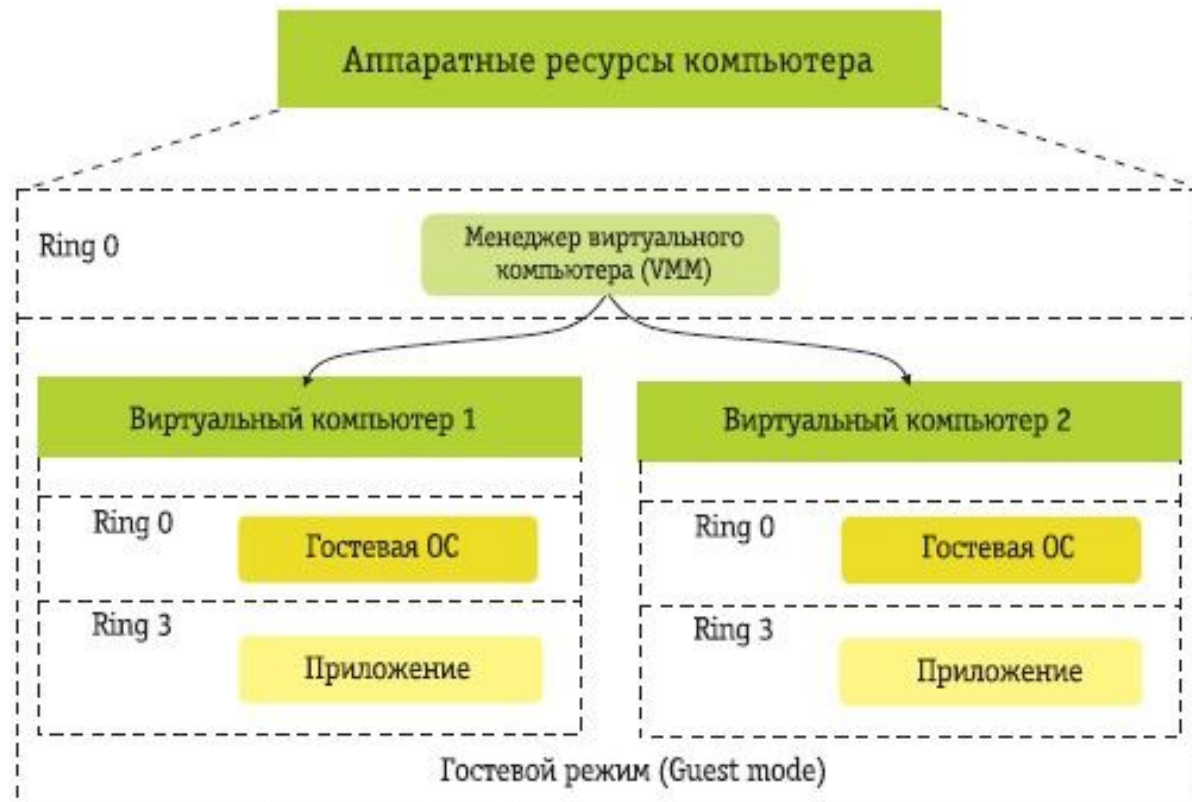
Аппаратная виртуализация AMD реализуется путем введения новых режимов работы процессора и дополнительных инструкций:

- Новый режим процессора: Guest Mode
- Новая структура данных: Virtual Machine Control Block (VMCB)
- Новая инструкция: VMRUN
- Новый режим памяти: Real Mode w/ Paging

AMD



AMD



Intel

Аппаратная виртуализация VT-х описывается специальной структурой VMCS (Virtual Machine Control Structure) и является небольшим участком физической оперативной памяти, хранящим:

- минимально необходимые данные для запуска гостевой ОС,
- данные, необходимые для безопасного выхода из режима работы гостевой ОС,
- некоторые настройки, относящиеся к управлению этой виртуальной машиной.

Принцип работы VT-x

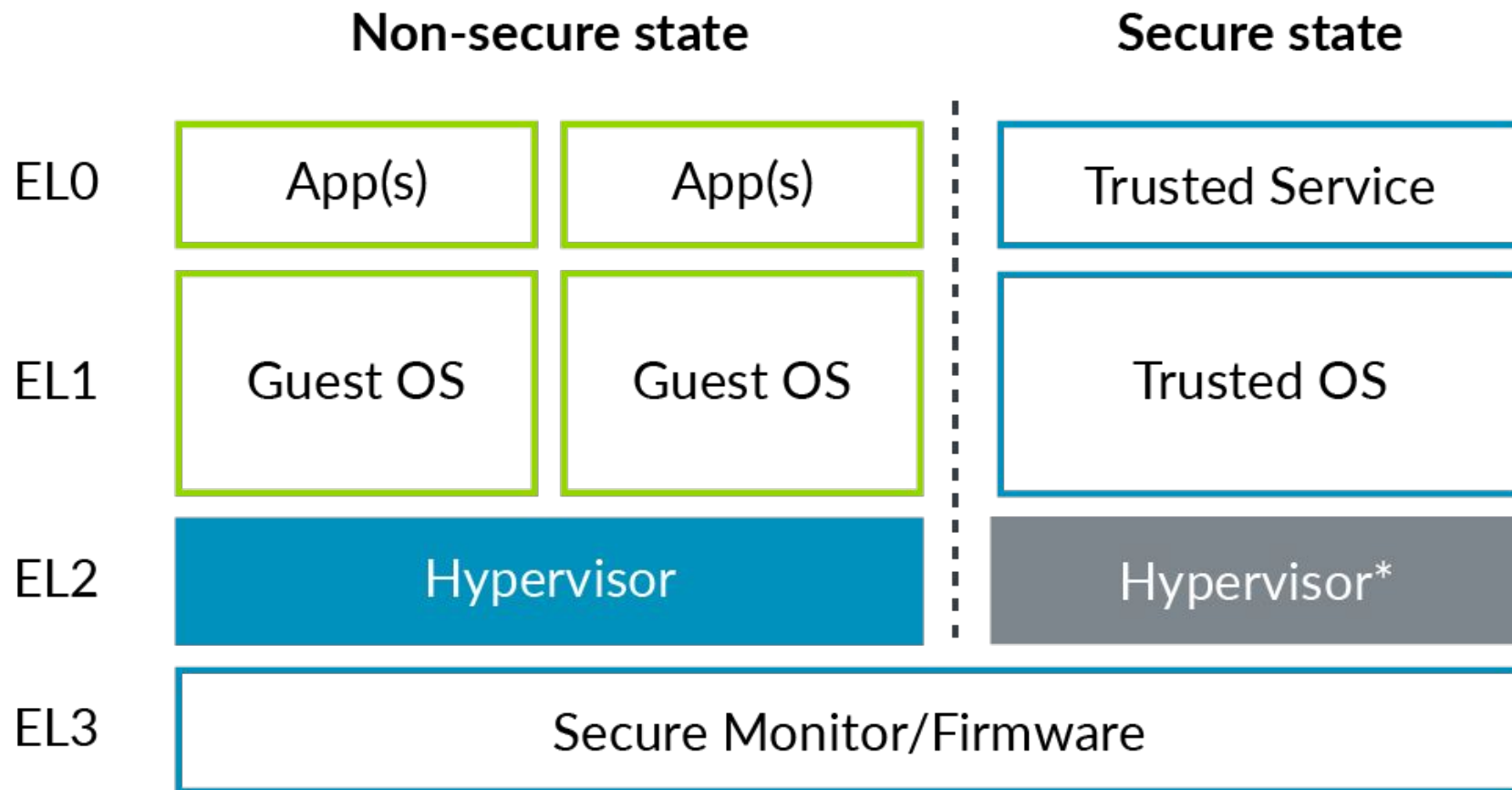




ARM

ARM гипервизор работает с уровнем исключения EL2. Только программное обеспечение, работающее на уровне исключения EL2 или выше, может получить доступ и настроить различные функции виртуализации.

ARM



ИСТОЧНИК



KVM



KVM

Для исполнения прямых аппаратных запросов в ОС должна иметься библиотека, которая направляла бы эти запросы аппаратной части напрямую.

Red hat создала ассоциацию Open Virtualization Alliance, которая была признана решить проблему отсутствия базового гипервизора для ядра Linux. Так и был создан гипервизор **KVM** или **Kernel-based Virtual Machine**.

Принцип KVM

Гипервизор KVM представляет из себя загружаемый модуль ядра Linux, который предназначен для обеспечения виртуализации на платформе Linux x86.

Сам модуль содержит:

- компонент собственно виртуализации (kvm.ko),
- процессорно-специфический загружаемый модуль kvm-amd.ko либо kvm-intel.ko.



Принцип KVM

Начиная с версии ядра 2.6.20, KVM является основной составляющей Linux. Иными словами, если у вас стоит Linux, то у вас уже есть KVM.

Необходимым условием для использования KVM является поддержка одной из инструкций виртуализации — Intel VT-x, AMD-V или ARM с поддержкой EL2



Принцип KVM

Сам по себе KVM не выполняет эмуляции. Вместо этого программа, работающая в пространстве пользователя, использует интерфейс `/dev/kvm` для настройки адресного пространства гостя виртуальной машины, через него же эмулирует устройства ввода-вывода и видеоадаптер.

Принцип KVM

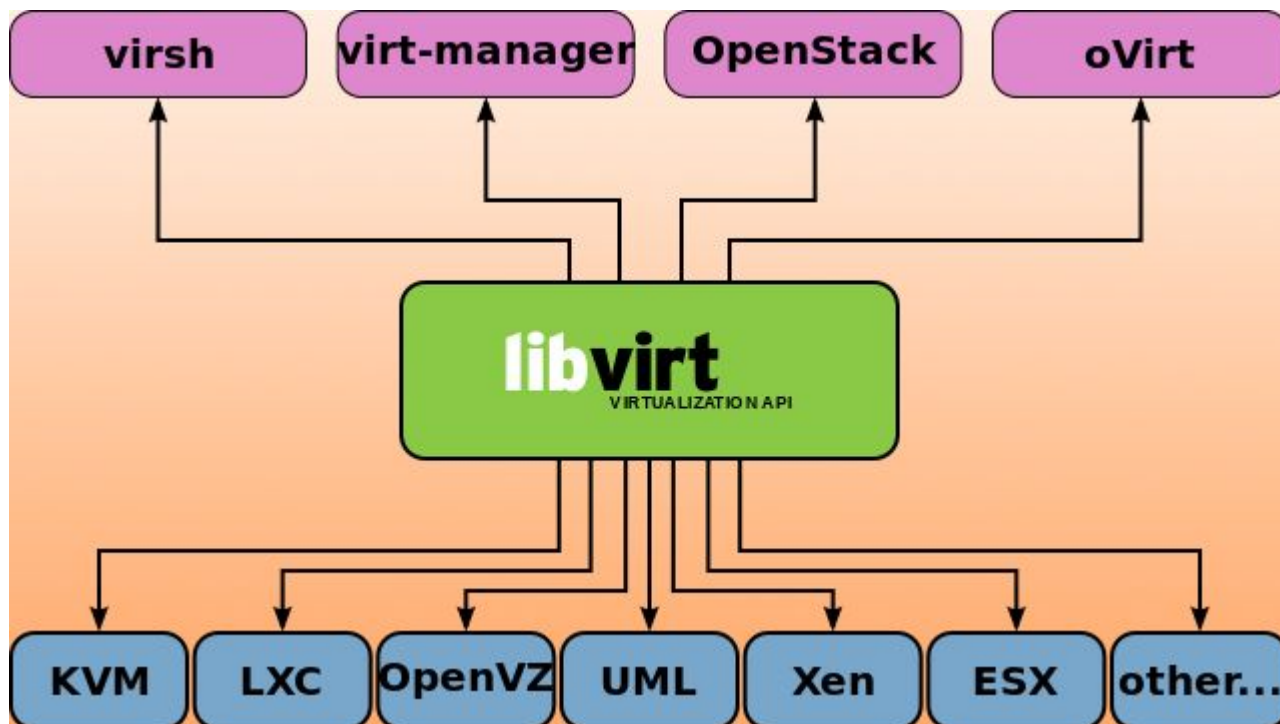
KVM позволяет виртуальным машинам использовать немодифицированные образы дисков QEMU, VMware и других, содержащие ОС.

Каждая виртуальная машина имеет свое собственное виртуальное аппаратное обеспечение:

- сетевые карты,
- диск,
- видеокарту,
- и другие устройства.

Использование KVM

Для наглядности рассматривается виртуализация KVM на базе библиотеки **virt-manager**





libvirt



libvirt

libvirt — это набор инструментов, предоставляющий единый API к множеству различных технологий виртуализации.

При использовании libvirt не важно какой «бекенд»: Xen, KVM, VirtualBox или что-то ещё. Более того, можно использовать libvirt внутри Ruby, Python, C++ и многих других программ Или удаленно подключаться к виртуальным машинам по защищенным каналам.

libvirt — это просто API, а вот как с ним взаимодействовать решать пользователю.

libvirt

Давайте попробуем.

Сначала проверим, поддерживается ли аппаратная виртуализация. На самом деле, работать будет и без её поддержки, только гораздо медленнее.

```
egrep --color=auto 'vmx|svm|0xc0f' /proc/cpuinfo
```

libvirt

Так как KVM — это модуль ядра Linux, то нужно проверить, загружен ли он уже, и если нет, то загрузить:

```
lsmod | grep kvm
```

Если модуль не загружен:

```
modprobe kvm
```

```
modprobe kvm_intel # или modprobe kvm_amd
```

libvirt

Возможна ситуация, что аппаратная виртуализация выключена в BIOS. Поэтому, если модули `kvm_intel/kvm_amd` не подгружаются, то необходимо проверить настройки BIOS.

Теперь установим необходимые пакеты. Проще всего сделать это, установив сразу группу пакетов:

CentOS :

```
yum group list «Virtual*»
```

```
[root@localhost etc]# yum group list "Virtual*"
Last metadata expiration check: 0:01:21 ago on Sun 16 May 2021 03:13:44 AM EDT.
Available Environment Groups:
  Virtualization Host
```


libvirt

Для управления виртуальными машинами из командой строки используется утилита `virsh`. Проверить, есть ли хотя бы одна виртуалка, можно командой:

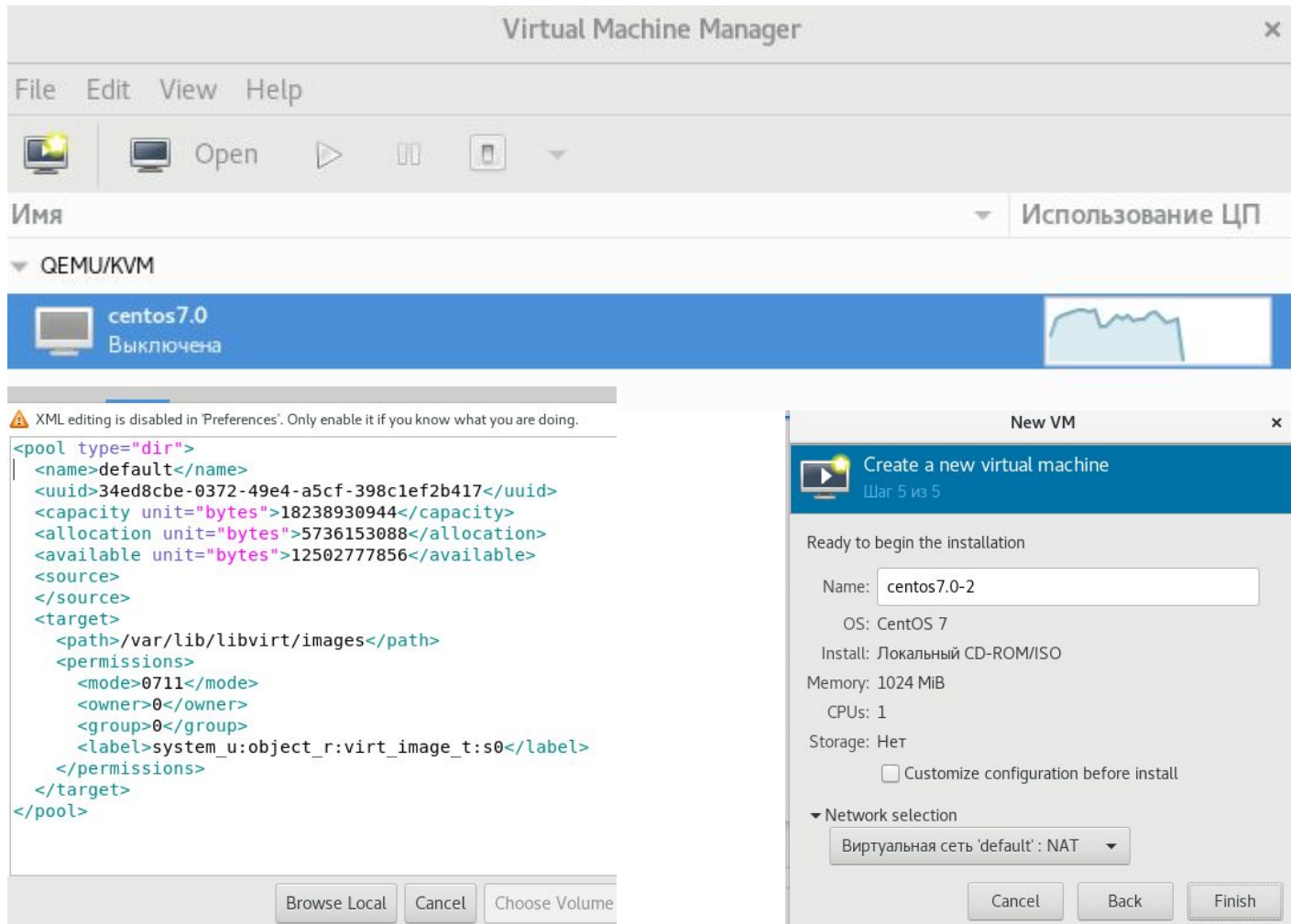
```
virsh list
```

Для использование GUI — `virt-manager`.

`virsh` умеет создавать виртуалки только из XML файлов, формат которых можно изучить в [документации libvirt](#).

```
[root@localhost etc]# yum install virt-manager_
```

libvirt



libvirt

Пример использования **libvirt**

```
virt-install --name mkdev-vm-0 \  
--location ~/Downloads/CentOS-7-x86_64-Minimal-20091.iso \  
--memory=1024 --vcpus=1 \  
--disk size=8
```

Присоединиться можно при помощи virt-manager.



VIRTUALIZATION GETTING STARTED GUIDE

Одно из преимуществ использования KVM/libvirt — потрясающая документация, в том числе создаваемая компанией Red Hat.

[Руководство по началу работы с виртуализацией](#)



XEN

XEN



Xen — кроссплатформенный гипервизор, разработанный в компьютерной лаборатории Кембриджского университета и распространяемый на условиях лицензии GPL.

Основные особенности:

- поддержка режима паравиртуализации помимо аппаратной виртуализации,
- минимальность кода самого гипервизора, за счёт выноса максимального количества компонентов за пределы гипервизора.

<https://xenproject.org>

HYPER-V



Microsoft Hyper-V

Microsoft

HYPER-V система аппаратной виртуализации для x64-систем на основе гипервизора. Все они позволяют виртуализировать серверные платформы x86-64 и представляют собой системы аппаратной виртуализации для VPS хостинга и не только.

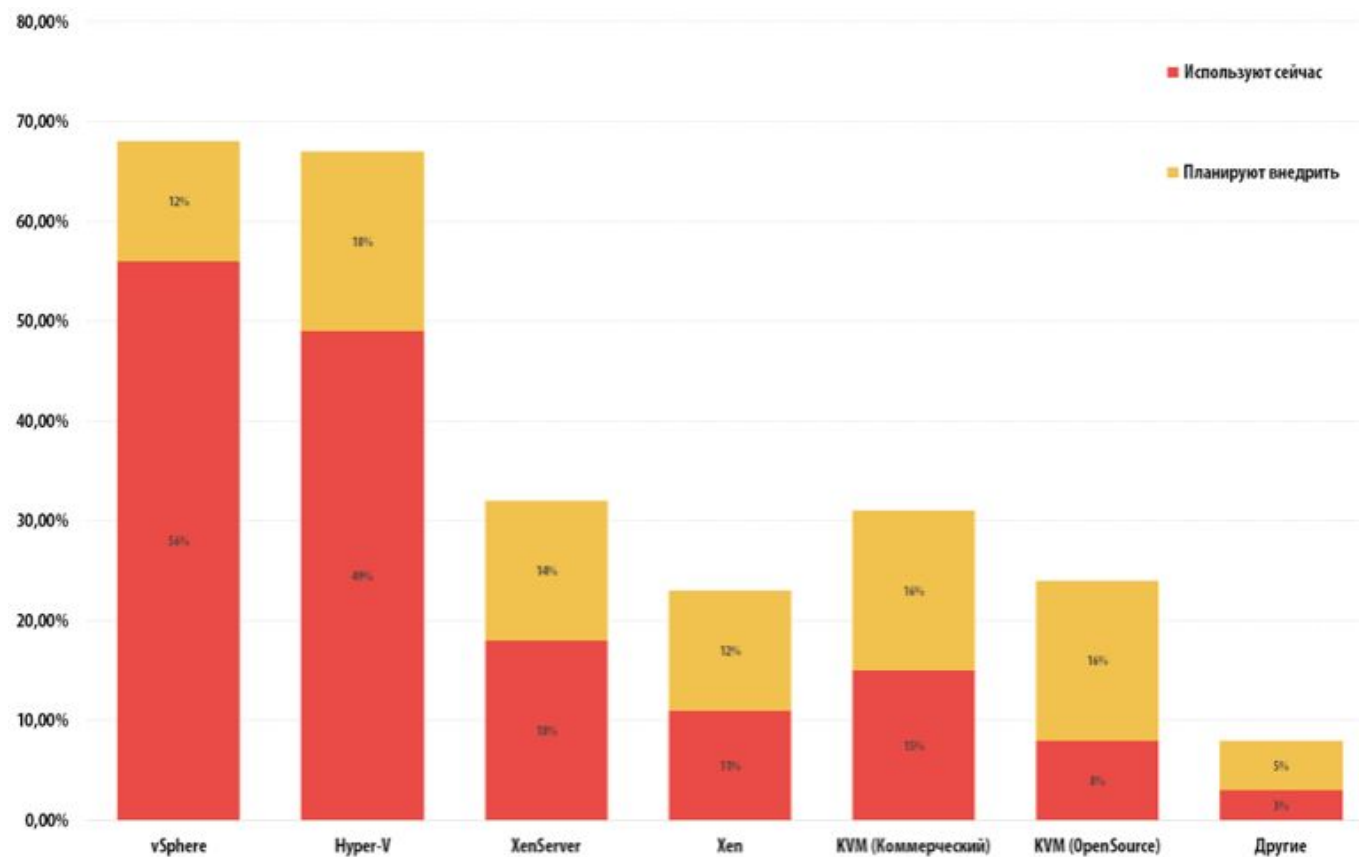
Windows-ая программа и с Win платформой работает на ура!

Для сравнения

	Best Value	Bare Metal	KVM	Xen
Timed MAFFT Alignment	lower	7.78	7.795	8.42
Smallpt	lower	160	162	167.5
POV-Ray	lower	230.02	232.44	235.89
PostMark	higher	3667	3824	3205
OpenSSL	higher	397.68	393.95	388.25
John the Ripper (MD5)	higher	49548	48899.5	46653.5
John the Ripper (DES)	higher	7374833.5	7271833.5	6911167
John the Ripper (Blowfish)	higher	3026	2991.5	2856
CLOMP	higher	3.3	3.285	3.125
C-Ray	lower	35.35	35.66	36.13
7-Zip	higher	12467.5	12129.5	11879

[Сравнение гипервизоров](#)

Динамика рынка



Источник



QEMU



QEMU

QEMU — свободная программа с открытым исходным кодом для эмуляции аппаратного обеспечения различных платформ.

Включает в себя эмуляцию процессоров Intel x86 и устройств ввода-вывода. Может эмулировать 80386, 80486, Pentium, Pentium Pro, AMD64 и другие x86-совместимые процессоры; ARM, MIPS, RISC-V, PowerPC, SPARC, SPARC64 и частично m68k.

<https://www.qemu.org>

Режимы QEMU

Полная эмуляция системы — полностью эмулирует устройство, включая все его компоненты, процессор и различные периферийные устройства. Он может использоваться для запуска нескольких ОС без перезагрузки или отладки системного кода.

Эмуляция пользовательского режима — работает только для Linux хоста, позволяет запускать процессы Linux, скомпилированные для одной архитектуры в другой, например, ARM программы в x86. Полезно для разработки, кросс-компиляции и отладки.

QEMU

Установка

Debian:

```
sudo apt install qemu-kvm qemu qemu-system
```

CentOS:

```
sudo yum install qemu-kvm qemu qemu-system
```

ArchLinux:

```
sudo pacman -i qemu-kvm qemu qemu-system
```

FreeBSD:

```
cd /usr/ports/emulator/qemu  
make && make install && make install clean
```

QEMU

Эмулятор qemu создает много команд, но их можно разделить на группы:

- qemu-архитектура — эмуляция окружения пользователя для указанной архитектуры;
- qemu-system-архитектура — эмуляция полной системы для архитектуры;
- qemu-img — утилита для работы с дисками;
- qemu-io — утилита для работы с вводом/выводом на диск;
- qemu-user — оболочка для qemu-архитектура, позволяет запускать программы других архитектур в этой системе;
- qemu-system — оболочка для qemu-system-архитектура, позволяет полностью эмулировать систему нужной архитектуры.

QEMU

Синтаксис команды такой:

```
$ qemu-system параметры
```

Куда сложнее здесь синтаксис каждого из параметров:

```
имя_параметра имя_опции=значение:значение2
```

QEMU

Мы рассмотрим только основные параметры, и их опции, которые нам понадобятся:

- `machine` указывает тип компьютера, который вы собрались эмулировать, можно выбрать `ubuntu`, `pc`, `pc-q35` и другие варианты, смотрите подробнее командой — `machine help`;
- `smp` — тип процессора, можно передать непосредственно тип процессора, а также дополнительные флаги;

Все остальные можно посмотреть тут:

https://wiki.qemu.org/Category:User_documentation

QEMU

Демонстрация создания виртуальной машины:

```
qemu-img create -f qcow2 test 1G
```

```
qemu-system-x86_64 -hda ubuntu.qcow -boot d -cdrom  
~/downloads/name_iso.iso -m 640
```



GNS 3

GNS3



Graphical Network Simulator — это графический симулятор сети, который позволяет смоделировать виртуальную сеть из маршрутизаторов и виртуальных машин. Незаменимый инструмент для обучения и тестов. Работает практически на всех платформах. Отлично подходит для создания стендов на десктоп машинах.



GNS3

В зависимости от аппаратной платформы, на которой будет использоваться GNS3, возможно построение комплексных проектов, состоящих из маршрутизаторов Cisco, Cisco ASA, Juniper, а также серверов под управлением сетевых операционных систем.

GNS3

The screenshot displays the GNS3 management console interface. The main workspace shows a network topology with three nodes: DebianFiltr-1 (a laptop icon), Switch1 (a switch icon), and pFsense-1 (a laptop icon). They are connected in a line: DebianFiltr-1 is connected to Switch1, which is connected to pFsense-1. The interface includes a top toolbar with various icons for file operations, simulation control, and editing. On the left is a vertical toolbar with icons for adding and managing nodes. On the right, there are two summary panels: 'Topology Summary' and 'Servers Summary'. The 'Topology Summary' panel lists the nodes and their console status. The 'Servers Summary' panel shows the status of the local host.

new1 - GNS3

Topology Summary

Node	Console
DebianFiltr-1	none
pFsense-1	none
Switch1	none

Servers Summary

- MacBook-Pro-Aleksandr.local CPU 55.7%, RAM 81.9%

Console

GNS3 management console.
Running GNS3 version 2.2.17 on Darwin (64-bit) with Python 3.6.5 Qt 5.11.0 and PyQt 5.10.1.
Copyright (c) 2006-2021 GNS3 Technologies.
Use Help -> GNS3 Doctor to detect common issues.



Итоги

Итоги

Сегодня мы рассмотрели:

- различные системы виртуализации,
- работу систем виртуализации на различных уровнях,
- возможность создания абстрактной модели.





Домашнее задание

Домашнее задание

Давайте посмотрим ваше [домашнее задание](#).

- Вопросы по домашней работе задавайте **в чате** мессенджера .
- Задачи можно сдавать **по частям**.
- Зачёт по домашней работе проставляется после того, как **приняты все задачи**.

**Задавайте вопросы и
пишите отзыв о лекции!**

Александр Зубарев