Операционные системы

Виртуализация и облако [1]

Соловьев А. В.

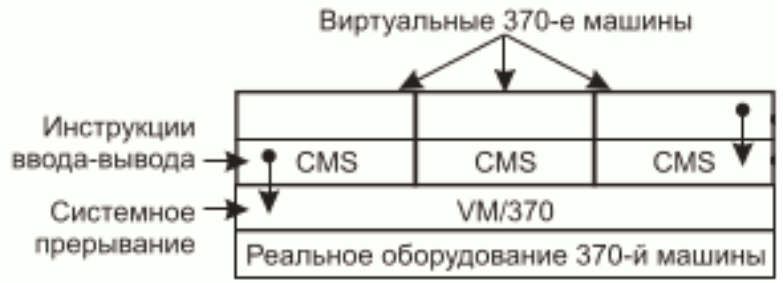
ПетрГУ – КИИСиФЭ

(Rev. 2018 06 07)

[1] По материалам «Таненбаум Э., Бос Х. Современные операционные системы. СПб.: Питер, 2015.»

История

1967, IBM, CP/CMS (исследовательский проект)
1972, VM/370 для семейства IBM System/370



Основа системы – *монитор виртуальных машин* – запускается непосредственно на обычном оборудовании и обеспечивает многозадачность, предоставляя верхнему уровню не одну, а несколько виртуальных машин (ВМ). Эти ВМ являются точной копией исходной аппаратуры (режим ядра и пользователя, устройства ввода-вывода, прерывания и др.)
На каждой ВМ способна работать любая ОС, которая может быть запущена непосредственно на самом оборудовании. На разных ВМ могут быть запущены разные ОС (OS/360 или CMS[2]).
В наст. время (с 2000) – z/VM для IBM zSeries.

На x86 первая коммерчески успешная виртуализация – VMware (1999).

[2] Conversational Monitor System – однопользовательская интерактивная система диалоговой обработки

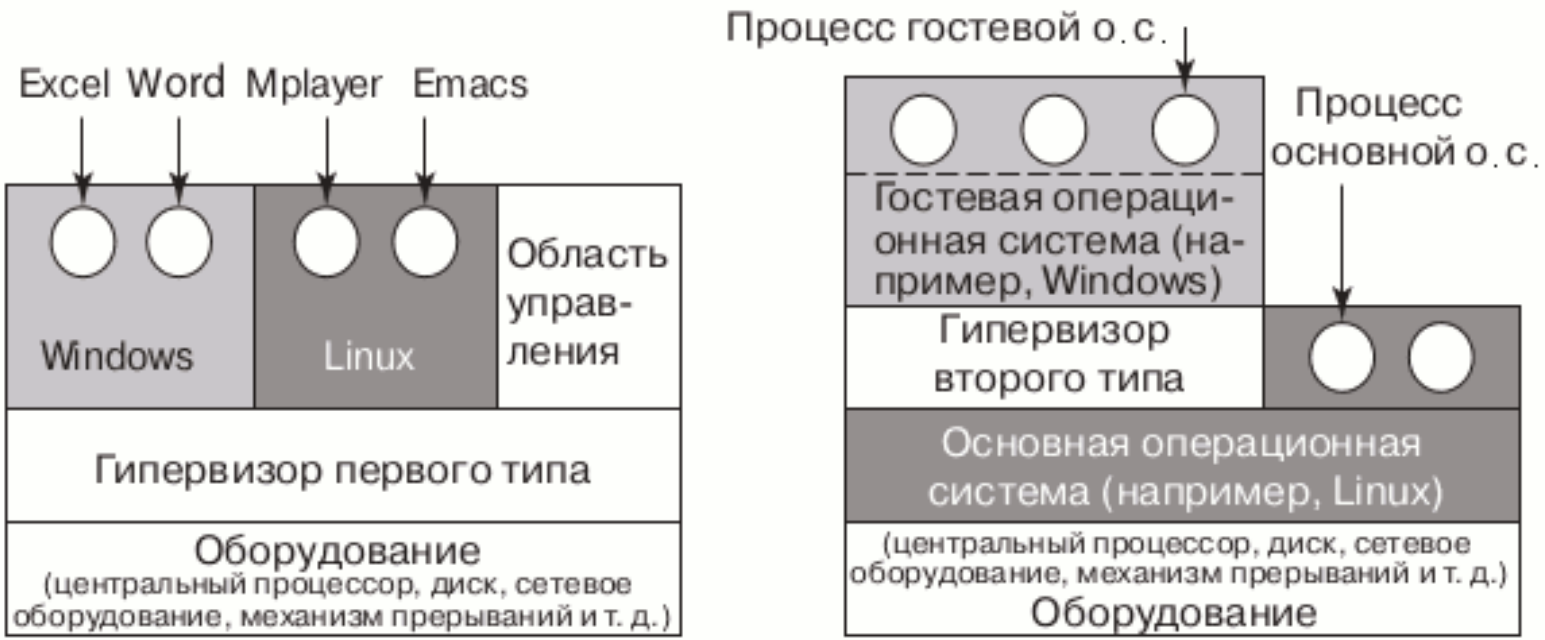
Требования, применяемые к виртуализации

1974, Gerald Popek & Robert Goldberg «Formal Requirements for Virtualizable Third Generation Architectures»

1. Безопасность (у гипервизора должно быть полное управление виртуализированными ресурсами).
2. Эквивалентность (поведение программы на ВМ и на реальном оборудовании должно быть идентичным).
3. Эффективность (основная часть кода в ВМ должна выполняться без вмешательства гипервизора).

Машина может быть подвергнута виртуализации, только если *служебные инструкции* являются поднабором *привилегированных инструкций*.
Служебные инструкции – инструкции, чувствительные к виртуализации (sensitive instructions): ввод-вывод, смена настроек MMU и т. п.
Привилегированные инструкции (privileged instructions) – инструкции, вызывающие системные прерывания.
Intel 386 не обладает этим свойством (дефект архитектуры): POPF (с битом запрета прерываний); MOV AX,CRx (чтение); сегментные регистры и CPL и т. п.
(В VMware проблема решалась за счёт двоичной трансляции).
2005, Intel – Virtualization Technology (VT), AMD – Secure Virtual Machine (SVM).

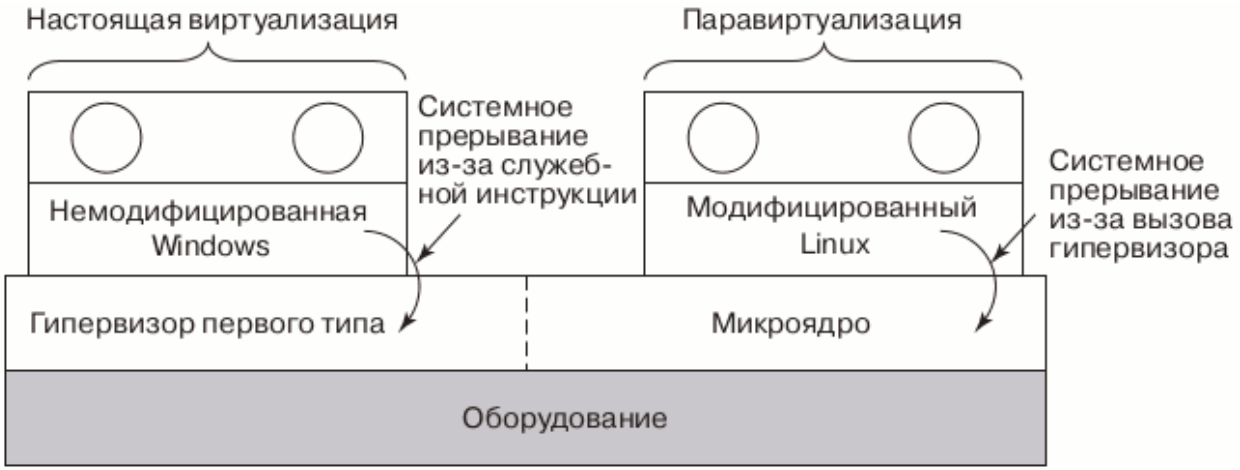
Гипервизоры первого и второго типа



Гипервозор I типа – единственная программа, запущенная в самом привилегированном режиме. Задача: поддержка нескольких копий имеющегося оборудования.

Гипервизор II типа – для создания процессов, сохранения файлов и т. д. используется основная ОС и её ФС. На практике для поддержки работы гипервизора требуется ещё модуль ядра.

Паравиртуализация

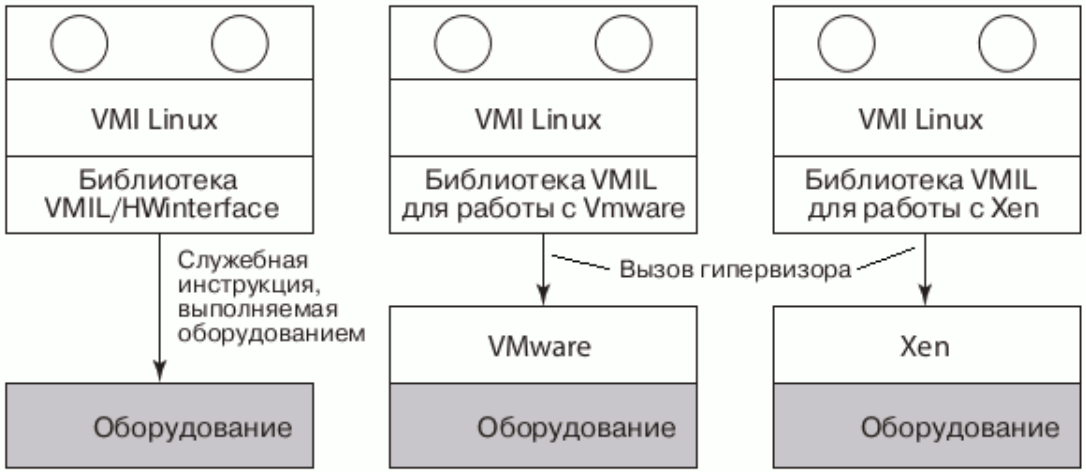


Проще?
Быстрее?

Паравиртуализация представляет машиноподобный программный интерфейс, который явно раскрывает факт наличия виртуальной среды (набор гипервызовов для привилегированных служебных операций типа обновления таблиц страниц).

2005–2011, VMware, Virtual Machine Interface (VMI)

2006, Xen Group (IBM, RedHat, VMware, Xen), “paravirt-ops” (pv-ops)



Примеры систем виртуализации

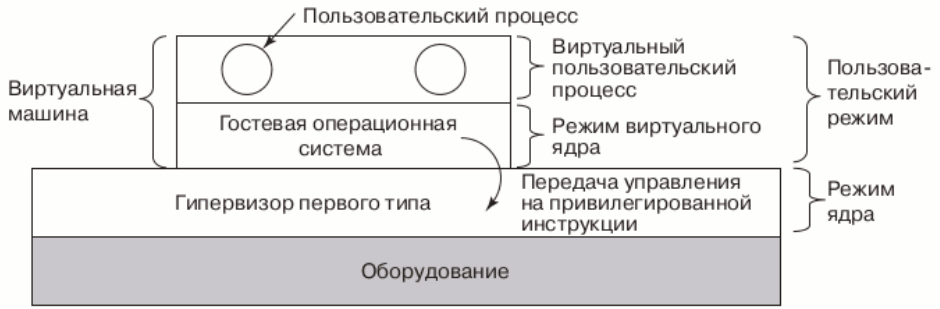
Полные системы виртуализации:

Название	Тип	VT/SVM	Примечание
VMware ESX Server	I	—	
VMware Workstation	II	—	
Xen (PV-режим)	I	+	паравиртуализация для Linux, NetBSD, FreeBSD
Xen (HVM-режим)	I	+	гибридный I, требует в dom0 Linux, OpenSolaris, OpenBSD или NetBSD
Hyper-V	I	+	элементы паравиртуализации, в т.ч. для Linux
KVM	II	+	на основе QEMU
VirtualBox	II	—	на основе QEMU

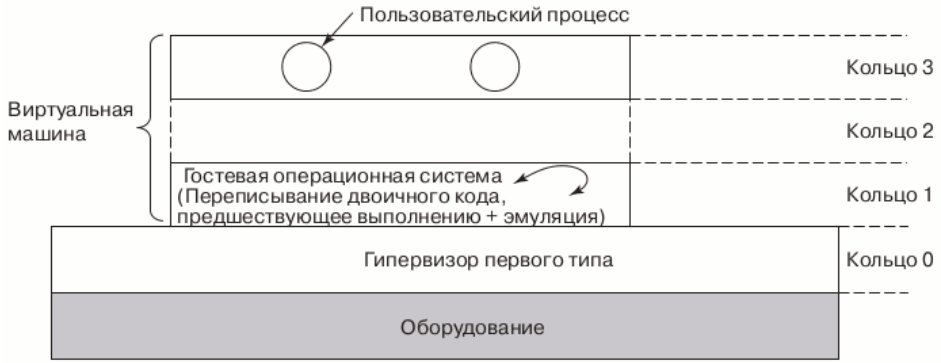
Виртуализация на уровне процесса (решает задачу запуска процесса, изначально написанного для другой ОС и/или архитектуры) – «эмулятор»:
Для ОС – например, WINE.
Для аппаратной архитектуры (двоичная трансляция) – например, QEMU.

Технологии эффективной виртуализации (1)

Виртуализация оборудования, поддерживающего VT/SVM



Виртуализация оборудования, не готового к виртуализации:



Техника двоичной трансляции:

Базовый блок (basic block) – небольшая прямая последовательность инструкций, завершающаяся переходом. То есть в базовом блоке нет переходов, вызовов, системных прерываний или других инструкций, вызывающих передачу управления (за искл. самой последней инструкции, которая именно это и делает).
Перед выполнением базового блока гипервизор сначала сканирует его, чтобы определить, не содержит ли он служебных инструкций и, если таковые имеются, заменяет их вызовом процедуры гипервизора, занимающейся их обработкой.
Переход в последней инструкции также заменяется вызовом, направленным в гипервизор (чтобы гарантировать возможность повторения процедуры для следующего базового блока).

Технологии эффективной виртуализации (2)

Всегда ли виртуализация на основе VT/SVM выигрывает у двоичной трансляции?

- Виртуализация на VT/SVM-процессорах приводит к выдаче большого количества системных прерываний, которые на современном оборудовании обходятся очень дорого, поскольку разрушают кэши ЦП, TLB-буферы и таблицы предсказания переходов.
- При двоичной трансляции служебные инструкции внутри выполняемого процесса заменяются вызовами процедур гипервизора, таких издержек от контекстного переключения не происходит. Код может быть успешно кэширован.
- При осуществлении двоичной трансляции сам оттранслированный код по сравнению с исходным кодом может быть более быстрым в выполнении. (Например, инструкция CLI может исполняться ЦП очень медленно, а на виртуальной машине это может быть реализовано в виде изменения обычной переменной (быстро). Тем не менее при использовании современного VT/SVM-ЦП аппаратное решение превосходит по эффективности программное).

Виртуализация памяти (1)

Для каждой ВМ гипервизору нужно создавать *теневую таблицу страниц* (shadow page table), отображающую виртуальные страницы, используемые ВМ, на реальные страницы, предоставляемые гипервизором.

- Либо при каждом изменении гостевой ОС своих таблиц страниц гипервизор должен также вносить изменения в теневую таблицу страниц.
- Либо гипервизор просто позволяет гостевой ОС добавить новое отображение к её таблицам страниц, как она того пожелает. Как только гостевая ОС пытается обратиться к любой из новых страниц, происходит ошибка отсутствия страницы и управление переходит к гипервизору.

Ошибки отсутствия страницы:

- *вызванные гостевой ОС* (guest-induced page faults) (должны быть снова введены в гостевую ОС).
- *вызванные гипервизором* (hypervisor-induced page faults) (для обновления теневых таблиц страниц).

Виртуализация памяти (2)

Аппаратная поддержка виртуальной памяти –
Second Level Address Translation (SLAT).
I уровень: гостевые виртульные адреса → гостевые физические адреса.
II уровень: гостевые физические адреса → машинные физические адреса.
2008, AMD, Rapid Virtualization Indexing (RVI), или Nested Page Tables (NPT)
2009, Intel, Extended Page Tables (EPT)

Возвращение памяти
Перерасход памяти (overcommitment) – гипервизор выделяет гостевым ОС больше памяти, чем физически имеется в машине.
ВМ могут совместно использовать страницы, имеющие одинаковое содержимое (*дедупликация*).
Раздувание (ballooning) – в каждую ВМ загружается модуль псевдодрайвера устройства, который общается с гипервизором. При «вздутии» такого модуля дефицит памяти в гостевой ОС растёт, и она будет реагировать выгрузкой тех страниц, которые считает наименее ценными. И наоборот, как только такой модуль «сдувается», у гостевой ОС появляется больше памяти для распределения.

Виртуализация ввода-вывода (1)

Проблема: что делать гипервизору, когда гостевая ОС пытается получить доступ к периферийному оборудованию?

Диски: Гипервизор может информировать гостевую ОС, что у него имеется обычный IDE-диск, и позволить гостевой ОС установить драйвер IDE-диска. Когда этот драйвер выдает команды управления IDE-диском, гипервизор преобразует их в команды управления имеющимся диском или преобразовывать их в обращение к файлу или разделу реального физического диска.

Сеть: гипервизор может выполнять роль виртуального коммутатора (каждой виртуальной машине назначается свой виртуальный сетевой интерфейс со своим MAC-адресом).

Виртуализация ввода-вывода (2)

- Гипервизоры II типа для вз.д. с устройствами I/O могут использовать основную ОС, её драйвера.
- Гипервизоры I типа (гибридные) могут иметь выделенную ВМ («домен 0»), чтобы использовать её ОС для вз.д. с устройствами I/O (эффективно при паравиртуализации).
- Либо гипервизор должен иметь собственные драйвера для оборудования.

Поддержка со стороны оборудования:

- I/O MMU использует таблицы страниц для отображения адреса памяти, которым желает воспользоваться устройство (адрес устройства), на физический адрес (прямая передача устройства (device pass through), изоляция устройств (device isolation), переназначение прерываний (interrupt remapping)).
- Single root I/O virtualization (SR-IOV) – виртуализация ввода-вывода в отдельно взятом физическом устройстве.

Virtual Appliance

«Виртуальный прибор» – образ предварительно сконфигурированной виртуальной машины, готовой к запуску под каким-либо гипервизором.
Цель VA – исключить затраты (времени и ресурсов) на инсталляцию, конфигурирование и поддержку какого-то сложного программного комплекса (например, IP-АТС + веб-сервер + система генерации речи + ...).
Как правило, VA собирают для установки какого-то одного определённого программного комплекса, при этом VA имеет веб-интерфейс для своего конфигурирования и более тонкой настройки.
VA распространяется в виде файла-пакета. Например, Open Virtualization Format (OVF) – разработан и поддерживается VMware, IBM, Microsoft, Xen и др. (имеется стандарт ANSI). Содержит описание необходимого оборудования, образы дисков, сертификаты и т. п.
Применение:

- Грид (быстрое создание грид-узлов, не зависящих от HW и SW).
- Как элемент облака.

Облака (1)

Характеристики (признаки) облачного сервиса (NIST):

- Самообслуживание по требованию (On-demand self-service). Пользователи должны иметь возможность получать ресурсы автоматически, без человеческого участия.
- Широкий (высокоскоростной) доступ по сети (Broad network access). Все эти ресурсы должны быть доступны по сети посредством стандартных механизмов, чтобы ими могли воспользоваться гетерогенные устройства.
- Объединение ресурсов в пул (Resource pooling) для обслуживания нескольких пользователей с возможностью динамического назначения и освобождения ресурсов.
- Быстродействующая эластичность (Rapid elasticity) – незамедлительное масштабирование при получении и освобождении ресурсов в соответствии с потребностями пользователей.
- Учтённые услуги (Measured service). Поставщик должен вести учёт потреблённых ресурсов тем способом, который соответствует типу заранее оговорённых облачных услуг.

Облака (2)

Варианты реализации:

- *Software as a Service (SaaS)*: Потребитель пользуется ПО, запущенным у провайдера. Потребитель не контролирует инфраструктуру (сервера, сеть, ОС, дисковое хранилище,...). Примеры: CRM, веб-почта, игры,...
- *Platform as a Service (PaaS)*: Потребитель развёртывает на ресурсах провайдера собственное или приобретённое ПО с использованием компиляторов, библиотек и утилит, поддерживаемых провайдером. Потребитель не контролирует инфраструктуру, но имеет доступ к среде исполнения, СУБД, веб-серверу (его настройкам), средствам разработки и т. п.
- *Infrastructure as a Service (IaaS)*: Потребитель арендует базовые вычислительные ресурсы: ЦП, дисковое пространство, сеть, – и может развёртывать произвольную ОС и прикладное ПО. (Реализуется виртуальными машинами).

Пример IaaS-облака: 2006, Amazon Elastic Compute Cloud (EC2), основано на Xen.

Виртуализация на примере VMware Workstation

Особенности платформы WINTEL (x86)

Другие платформы (например, IBM zSeries) *вертикально интегрированы* – один и тот же производитель разработал оборудование, гипервизор, ОС и прикладное ПО. Для платформы WINTEL (x86) виртуализация сначала должна быть внедрена без поддержки любого из этих игроков в мире индустрии:

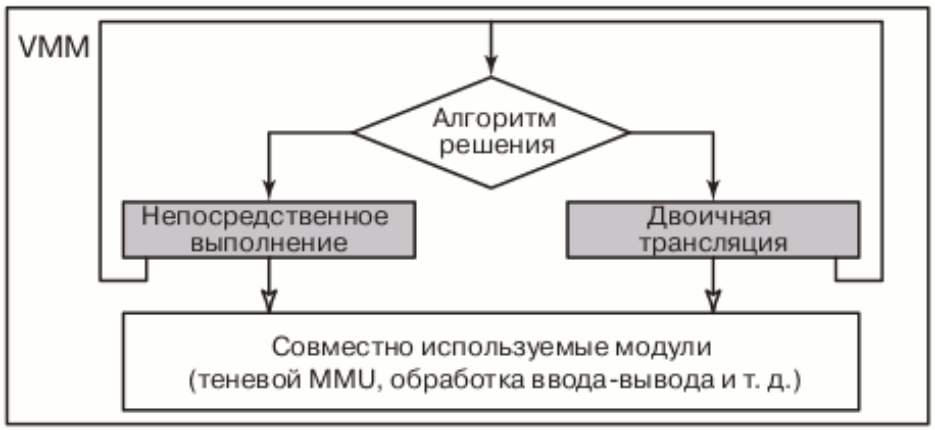
- Intel и AMD делают процессоры.
- Microsoft предлагает Windows (остальные ОС – не более 10 %).
- Третья группа компаний создает устройства ввода-вывода и периферийные устройства, а также соответствующие драйверы устройств.
- Системные интеграторы (HP, Dell, Sony, ...) собирают компьютерные системы для розничной продажи.

Проблемы архитектуры x86:

- Архитектура x86 была невиртуализируемой (с точки зрения критериев П&Г).
- Архитектура x86 была слишком сложна (CISC, 4 режима процессора: RM, PM, V86, SM, – с разной схемой работы сегментации и страничной трансляции).
- У машин x86 были разные периферийные устройства.
- Нужна модель, не требующая особого пользовательского опыта (классические гипервизоры устанавливались на производстве аналогично прошивкам).

VMM – Virtual Machine Monitor

VMM отвечает за выполнение инструкций виртуальной машины.



Двоичная трансляция выполняется:

- ВМ в данный момент запущена в режиме виртуального ядра;
- ВМ может заблокировать прерывания и выдать инструкции ввода-вывода (если IOPL == CPL);
- ВМ в данный момент запущена в RM, кроме того, для BIOS используется 16-битный режим.

VMM использует теневые таблицы страниц и дескрипторов сегментов, гарантирующие, что MMU может использоваться напрямую (без эмуляции).
VMM резервирует для себя верхние 4 Мб адресного пространства.
Код виртуального режима ядра исполняется VMM в кольце 1.
Поддерживается ограниченный набор гостевых операционных систем.

Платформа виртуального оборудования

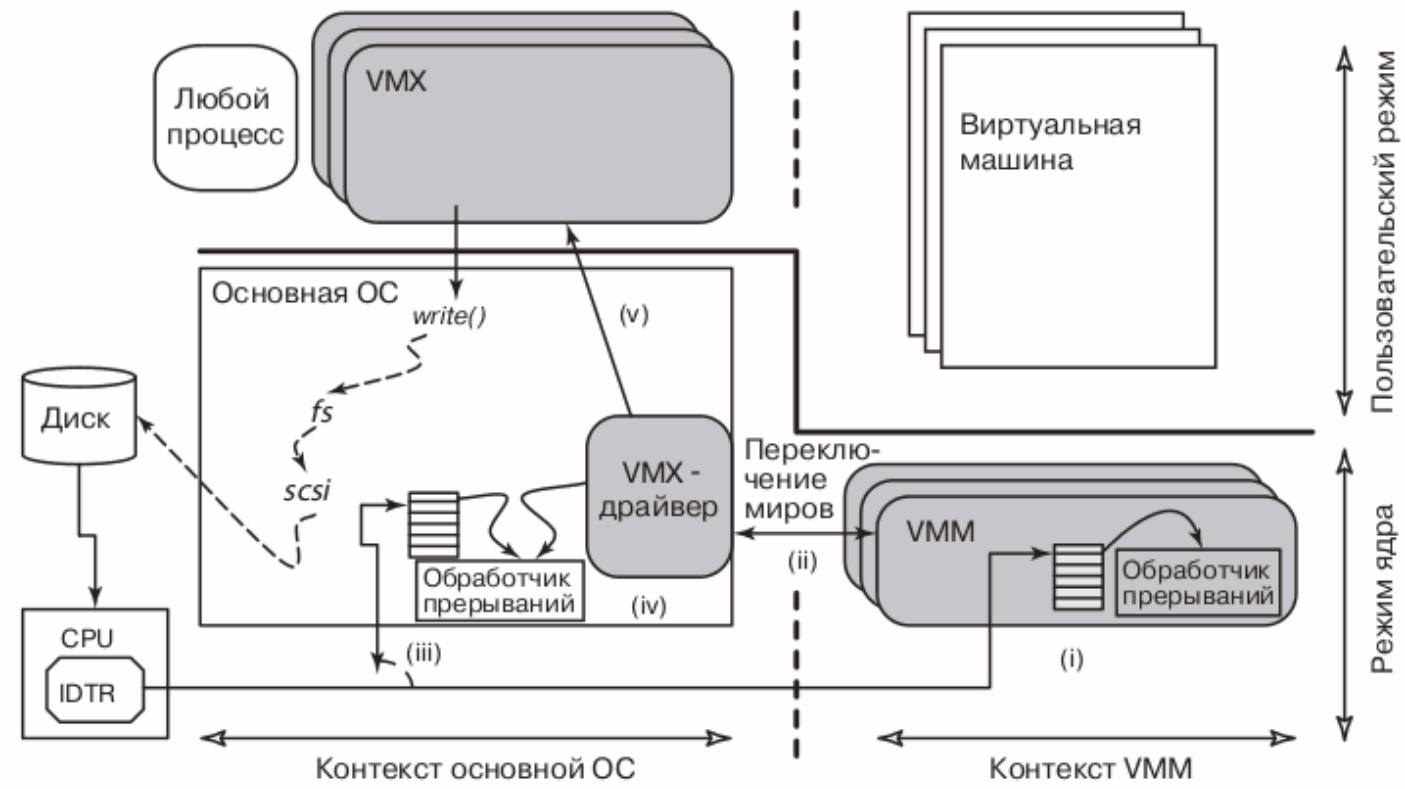
Мультиплексированное оборудование:

- Виртуальный ЦП x86 такой же, что и ЦП основного оборудования. Работа регламентировалась основной ОС.
- ОЗУ – распределялась и управлялась основной ОС (постранично).

Эмулируемое оборудование:

- Полностью эмулируемая совместимая шина PCI.
- Виртуальные диски: 4x IDE-диска или 7x Buslogic SCSI-диска, – в виде файлов или непосредственный доступ к заданному простому устройству.
- IDE-привод компакт-дисков, приводы флоппи-дисков – ISO-образы или эмулируемый доступ к реальному приводу.
- Графическая карта VMware обеспечивает запуск ВМ в окне или в полноэкранном режиме с поддержкой VGA и SVGA (нужен гостевой драйвер).
- Порты: COM1, COM2, LPT, – физические порты хост-машины или файл.
- Клавиатура и мышь – полностью эмулировались; события кода клавиши генерировались после их получения приложением VMware.
- Ethernet-карты (до трёх) – AMD Lance-совместимые (режим моста, NAT или только сеть с хост-машиной).
- Звуковая карта типа Soundblaster – полностью эмулировалась.

Компоненты VMware



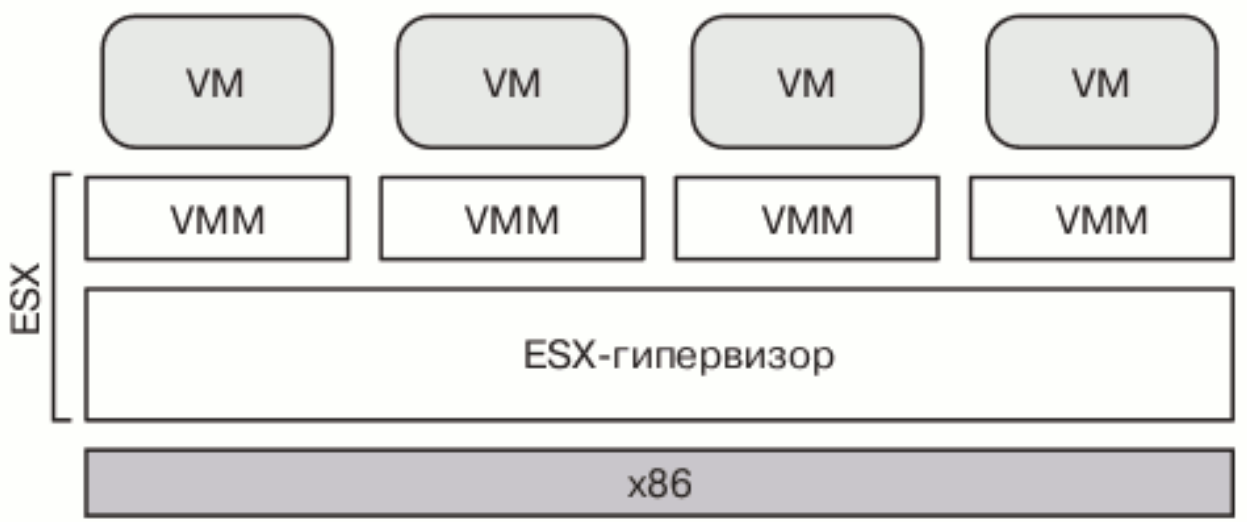
VMX (Virtual Machine Executable) – пользовательский интерфейс, эмуляция интерфейсных устройств, вз.д. с основной ОС (отдельный поток для каждой ВМ).
VMX-драйвер – организация «переключения миров» для запуска VMM.
VMM (Virtual Machine Monitor) – мультиплексирование ЦП, ОЗУ, обработчики исключений, двоичный транслятор, ... Работает в режиме ядра, но не в контексте основной ОС. Для каждой ВМ – свой экземпляр VMM.

«Переключение миров» (world switch)

Обработка аппаратного прерывания в режиме VMM:

1. У VMM нет драйверов устройств, обработка HW INTR невозможна.
2. Производится «переключение миров» обратно в VMX-драйвер.
3. VMX-драйвер эмулирует аналогичное прерывание в контексте основной ОС.
4. Выполняется стандартный обработчик основной ОС.
5. Диспетчер процессов основной ОС возвращает управление какому-либо процессу основной ОС или процессу VMX, который специальным вызовом VMX-драйвера может снова выполнить «переключение миров».

VMware ESX Server: гипервизор I типа



ESX (Elastic Sky X) заменяет функции основной ОС: планировщик задач, подсистема ввода-вывода, – оптимизированные под запуск ВМ (не требуется «переключения миров»).
ESX можно запустить только на ограниченном количестве серверных платформ, для которых у него есть драйвера.
VMFS – специальная файловая система для хранения образов ВМ и обеспечения высокой пропускной способности системы ввода-вывода.
VMotion – миграция работающей ВМ с одного ESX-сервера на другой.