Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

НА ТЕМУ:

«Классификация методов виртуализации программного обеспечения»

Студент ИУ7-13М (Группа) _____ (Подпись, дата) Мицевич М. Д. (И. О. Фамилия)

Руководитель НИР _____ (Подпись, дата) Тассов К. Л. (И. О. Фамилия)

2023 г.

РЕФЕРАТ

Расчетно-пояснительная записка к научно-исследовательской работе содержит 16 страниц, 0 иллюстраций, 1 таблица, 8 источников.

Научно-исследовательская работа представляет собой изучение предметной области виртуализации, описание основных методов, а также преимуществ и недостатков каждого из них. Рассмотрены различные подходы виртуализации программного обеспечения. Представлено описание методов полной, неполной, аппаратной, контейнерной и паравиртуализации. Проведено сравнение контейнеров и виртуальных машин.

Ключевые слова: виртуализация, гипервизор, виртуальная машина, контейнер, гостевая операционная система.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Многим организациям для поддержания процесса своей работы требуется несколько серверов с запущенными на них различными приложениями. К примеру, в компании может быть почтовый сервер, файловое хранилище, веб сервис. Каждый из этих серверов должен запускаться на определенной операционной системе.

Запуск каждого сервиса на отдельной физической машине влечет за собой большие расходы для компании по сравнению с запуском всего на одном сервере [4].

Одним из вариантов решения данной проблемы является использование технологии виртуализации, которая позволяет запускать несколько виртуальных машин на одной физической. Преимуществом такого подхода является то, что авария одной виртуальной машины не приводит к аварии на других.

Также виртуализация позволяет запускать устаревшие приложения на операционных системах, которые больше не поддерживаются на текущем физическом оборудовании.

Целью данной работы является классификация методов виртуализации программного обеспечения.

Для достижения поставленной цели требуется выполнить следующие задачи:

— провести анализ предметной области виртуализации;

— сформулировать критерии сравнения методов;

— провести сравнительный анализ методов виртуализации.

1 Анализ предметной области

1.1 Виртуализация

Под виртуализацией понимается предоставление набора вычислительных ресурсов или их логического объединения, абстрагированное от аппаратной реализации и обеспечивающее при этом логическую изоляцию вычислительных процессов, выполняемых на одном физическом ресурсе [1].

Критерии виртуализируемости системы впервые были сформулированы в статье [2]. В данной статье выделяется три основные критерия, которым должен отвечать монитор виртуальных машин:

— изоляция – каждая виртуальная машина должна быть изолирована и иметь доступ только к назначенным ей ресурсам. Она не должна иметь возможности влиять на работу монитора или других виртуальных машин;

— эквивалентность – любая программа, которая выполняется на виртуальной машине, должна демонстрировать поведение, полностью идентичное ее выполнению на реальной системе. Единственные различия могут быть связаны с доступностью ресурсов (например, может быть ограничен объем доступной памяти для виртуальной машины) и длительностью операций (из-за возможности разделения времени выполнения с другими виртуальными машинами);

— эффективность – статистически преобладающее подмножество инструкций виртуального процессора должно исполняться напрямую хозяйским процессором, без вмешательства монитора виртуальных машин.

В зависимости от способа реализации виртуализацию подразделяют [3] на:

— полную – эмулируются все инструкции гостевой операционной системы;

— неполную – нет сокрытия центрального процессора;

— паравиртуализацию – используются специальные операционные системы, которые знают, что они запускаются как гостевые;

— аппаратную виртуализацию – для виртуализации используются специальные инструкции процессора;

— контейнеризация – процесс запускается в специальном изолированном пространстве и использует то же ядро, что и основная операционная система.

1.2 Полная виртуализация

Первые гипервизоры полностью эмулировали базовое оборудование, определяя виртуальные замены всех основных вычислительных ресурсов: жестких дисков, сетевых устройств, прерываний, аппаратных средств материнской платы, BIOS. Такой режим позволяет запускать любые гостевые операционные системы без изменений и называется полной виртуализацией.

Основным недостатком такого типа виртуализации является снижение производительности системы, связанное с тем, что гипервизор должен постоянно выполнять трансляцию между реальным и виртуальным оборудованием системы [3].

Основным преимуществом полной виртуализации является независимость между платформой, под которую была разработана гостевая операционная система, и платформой, на которой она будет запускаться.

1.3 Неполная виртуализация

Все центральные процессоры с режимом ядра и пользовательским режимом имеют набор инструкций, ведущих себя по-разному в зависимости от того, в каком режиме они выполняются, в режиме ядра или в пользовательском режиме. В их число входят инструкции, осуществляющие ввод-вывод, изменяющие настройки блока управления памятью, такие инструкции называются служебными. Есть также набор инструкций, которые при выполнении в пользовательском режиме вызывают системные прерывания. Такие инструкции называются привилегированными.

Машина может быть подвергнута аппаратной виртуализации, только если служебные инструкции являются поднабором привилегированных инструкций [4]. Данное ограничение связанно с тем, что гипервизору для корректной работы гостевой операционной системы необходимо перехватывать от нее и обрабатывать все служебные и привилегированные инструкции.

Гипервизоры, которые выполняют неполную виртуализацию, переписывают часть кода на лету, заменяя проблемные инструкции безопасной кодовой последовательностью, эмулирующей исходную инструкцию. Перезапись полезна для замены инструкций, являвшихся служебными, но не входивших в число привилегированных. Такая технологий называется двоичной трансляцией [4].

Переписывать абсолютно все служебные инструкции нет необходимости. В частности, пользовательские процессы на гостевой операционной системе могут выполняться без модификации. Для служебных инструкций, входящих в состав привилегированных используется схема, при которой вызывается системной прерывание и обработчик гипервизора, который его обрабатывает. Обычно для реализации этого правила в гипервизорах имеется модуль, которые выполняется в пространстве ядра операционной системы, перенаправляющих системные прерывания к своим обработчикам.

Для реализации данного типа виртуализации для процессора x86 многие компании использовали следующий подход: гипервизор запускался на нулевом кольце защиты, все пользовательские приложения – на третьем, а гостевая операционная система – на первом. Таким образом, попытка доступа к памяти ядра гостевой операционной системы из пользовательского приложения приведет к нарушению прав доступа, а исполнение привилегированных инструкций гостевой операционной системы приведет к вызову системного прерывания с передачей управления гипервизору. Для обработки служебных инструкций в коде операционной системе используется динамическая двоичная трансляция, которая заменяет их на вызовы процедур гипервизора.

1.4 Паравиртуализация

Паравиртуализация – это техника виртуализации, при которой гостевая операционная система подготавливается к исполнению в виртуализированной

среде, для чего ее ядро незначительно модифицируется [5]. При паравиртуализации привилегированные инструкции, исполнение которых запрещено в гостевом режиме, подменяются на гипервызовы – прямые передачи управления гипервизору. Достойной производительности гостевых систем позволяет добиться характерная паравиртуализации оптимизация: замена одним гипервызовом нескольких последовательных привилегированных инструкций. Однако применение паравиртуализации ограничивается возможностью модификации кодов операционной системы.

Определенные паравиртуализованные функции могут быть реализованы и в гипервизорах с полной виртуализацией с использование специальных виртуальных драйверов в гостевых операционных системах, которые общающются напрямую с гипервизором.

Главным преимуществом такого подхода является более высокая скорость работы по сравнению с полной и неполной виртуализацией, так для работы гостевой операционной системы не требуется полная эмуляция всех инструкций и динамическая трансляция кода операционной системы.

Главным недостатком такого подхода является необходимость модификации кода гостевой операционной системы.

1.5 Аппаратная виртуализация

В 2004 и 2005 годах компаниями Intel и AMD были представлены функции процессора, которые способствуют виртуализации на платформе x86. В этой схеме центральный процессор и контроллер памяти виртуализируются аппаратно, хотя и под управлением гипервизора. Виртуализация, при которой используются специальные инструкции процессора называется аппаратной [6]. Производительность при этом выше, чем при неполной виртуализации, а гостевые операционные системы не должны знать, что они виртуализированы.

Основной замысел технологии заключается в том, что гостевая операционная система запускается в контейнере и работает в нем до тех пор, пока ее не будет вызвано исключение или не будет осуществлено системное прерывание в гипервизоре. Набор операций, вызывающих системное прерывание задается гипервизором.

1.6 Типы гипервизоров

Выделяют два основных типа гипервизоров:

— первый тип – гипервизор запускается без основной операционной системы и сам выполняет ее функции;

— второй тип – гипервизор запускается внутри основной операционной системы.

В первом случае гипервизор – единственная программа, которая запущена в привилегированном режиме.

Гипервизоры первого типа используются предприятиями, а второго удобны пользователям для быстрого развертывания тестового стенда.

1.7 Контейнерная виртуализация

Контейнеризация – это другой подход к изоляции который не использует гипервизор. Вместо этого он опирается на возможности ядра, изолирующие процессы от остальной системы. Каждый процесс имеет собственную корневую файловую систему и пространство имен процессов. Содержащиеся в нем процессы совместно используют ядро и другие сервисы основной операционной системы, но они не могут получить доступ к файлам или ресурсам за пределами своих контейнеров. Поскольку данная технология не требует виртуализации аппаратного обеспечения, ресурсные накладные расходы при виртуализации на уровне операционной системы меньше по сравнению с другими подходами.

Контейнеры представляют собой объединение множества существующих функций ядра, возможностей файловой системы и сетевых абстракций. Контейнерный движок – это управляющее программное обеспечение, которое управляет работой контейнеров.

Для изоляции процессов контейнерных движок использует следующие функции ядра [7]:

— пространства имен – задают, какие ресурсы ядра видны процессу;

— контрольные группы – выделяет и ограничивает ресурсы, такие как процессор, память, сетевой ввод-вывод, которые используются контейнерами;

— многослойные файловые системы – состоят из нескольких слоев.

Образы контейнеров представляют собой многослойную файловую систему, которая по своей организации напоминают корневую файловую систему дистрибутива Linux.

При создании контейнера в файловую систему образа добавляется еще один слой доступный как для чтения, так и для записи. При попытке изменения контейнером какого-либо файла в слое доступном только для чтения срабатывает стратегия copy on write и этот файл переносится в слой для записи, созданный при запуске контейнера.

2 Сравнение существующих решений

2.1 Сравнение контейнеров и виртуальных машин

Обе эти технологии создают иллюзию того, что на одной машине можно запускать несколько машин. Все эти машины, работающие под управлением основной машины, должны быть изолированы друг от друга, а также от основной машины. Разница заключается в том, как обе эти технологии способны достичь изоляции между различными машинами. Разница заключается в том, что контейнеры обычно выполняются на хостовой операционной системе, а виртуальные машины – на гипервизоре. Контейнерный движок обычно совмещен с ядром хостовой операционной системы.

Виртуальные машины и контейнеры отличаются тем, чем может быть гостевая операционная система. Виртуальные машины позволяют запускать гостевое ядро, отличное от ядра основной операционной системы. Это невозможно в случае контейнеров, поскольку ядро должно быть общим.

Контейнеры по сравнению с виртуальными машинами требуют меньше ресурсов [8]. Благодаря этому время запуска контейнеров меньше, чем у виртуальных машин. Репликация также проще в контейнерах, поскольку они не требуют отдельной операционной системы.

Безопасность контейнеров ниже, чем у виртуальных машин, так как все контейнеры разделяют одно ядро, в то время как виртуальные машины работают под управлением отдельной операционной системы, благодаря чему они могут использовать свои собственные функции безопасности и ядра.

2.2 Сравнение методов виртуализации

Главным преимуществом полной виртуализации является возможность запуска гостевой операционной системы разработанной под платформу отличную от той, где запущен гипервизор. Главным недостатком такого подхода являются накладные расходы на эмуляцию каждой инструкции. Полная виртуализация может быть полезна, к примеру, для запуска приложений, разработанных под операционную систему Windows и процессор x86, на операционной

системе Эльбрус, запущенной на одноименном процессоре.

При неполной виртуализации повышается производительность системы, так как эмулируются не все инструкции, а только служебные и привилегированные. При таком подходе необходимо, чтобы платформа, под которую разработана гостевая операционная система, совпадала с той, на которой запущен гипервизор.

Паравиртуализация позволяет запускать специальные гостевые операционные системы, которые знают, что они вирутализированы, и используют специальные гипервызовы для обращения к гипервизору. Накладные расходы на виртуализацию при таком подходе меньше чем при неполной виртуализации, но требуется модификация гостевых операционных систем.

При аппаратной виртуализации гипервизор для создания и управления виртуальными машинами использует специальные инструкции процессора. Производительность системы при таком подходе выше чем при неполной виртуализации, а для запуска виртуальной машины не требуется модификация гостевой операционной системы.

При контейнеризации отдельные процессы запускаются в изолированном пространстве и разделяют одно ядро основной операционной системы. При таком подходе снижаются накладные расходы на виртуализацию, и время на запуск. Недостатком контейнеризации по сравнению с виртуальными машинами является меньшей уровень безопасности и невозможность запуска контейнеров с ядром, отличающимся от основной операционной системы.

Сравнение методов виртуализации представлено в таблице 2.1.

Таблица 2.1 – Сравнение методов виртуализации

	Полная	**Неполная**	**Паравиртуализация**	**Аппаратная**	**Контейнерная**
Возможность запуска на другой архитектуре	+	-	-	-	-
Любая гостевая ОС	+	+	-	+	-
Создание без основной ОС	+	+	+	+	-
Создание без специальных инструкций процессора	+	+	+	-	+
Запуск как обычного пользовательского процесса	-	-	-	-	+

ЗАКЛЮЧЕНИЕ

Технология виртуализации позволяет запускать несколько виртуальных машин на одном физическом сервере.

В случае когда платформа, под которую разработана гостевая операционная система, отличается от плафтормы сервера, где она будет запускать, требуется использовать гипервизоры с полной виртуализацией.

Для решения задачи изоляции процессов, которые используют одно ядро операционной системы, но разные окружения, лучше всего подойдет технология контейнеризации.

В данной научно-исследовательской работе был:

— проведен анализ предметной области виртуализации;

— сформулированы критерии сравнения методов;

— проведен сравнительный анализ методов виртуализации.

В рамках работы были выполнены все поставленные задачи. Цель работы была достигнута.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Алексеев А. Л.* Гипервизоры и виртуальные машины // ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В НАУКЕ, ПРОМЫШЛЕННОСТИ И ОБРАЗОВАНИИ. — 2019. — С. 121—128.
2. *Popek J. G.* Formal requirements for virtualizable third generation architectures // Communications of the ACM. — 1974. — № 17.
3. *Nemeth E.* UNIX AND LINUX SYSTEM ADMINISTRATION HANDBOOK. — 2018.
4. *Таненбаум Э.* Современные операционные системы. — 2015.
5. *Рыбаков С. А.* Виртуализация ОС Эльбрус // Вестник Концерна ВКО Алмаз-Антей. — 2021. — 4 (39). — С. 67—75.
6. *Гоношенко С. В.* АППАРАТНО-ПРОГРАММНЫЕ СРЕДСТВА ВИРТУАЛИЗАЦИИ // Информационные технологии XXI века. — 2020. — С. 236—241.
7. *Shashank M. J.* Linux Containers and Virtualization. — Springer, 2020.
8. *Anuj K.* Docker containers versus virtual machine-based virtualization. — Springer, 2019. — С. 141—150.