Symbian обеспечивает стандартные LDD для периферийных устройств (драйвера медиа-устройств, контроллер USB и последовательные устройства связи). Однако изготовители телефонов часто разрабатывают собственные интерфейсы для специальных аппаратных средств, что требует разработки дополнительных LDD.

PDD управляет конкретным периферийным устройством от имени его LDD и содержит код, зависимый от устройства. PDD взаимодействует только с соответствующим ему LDD, таким образом приложение пользовательской стороны не может получить доступ непосредственно к PDD. На PDD возлагают обязанности взаимодействия с вариантными и простыми расширениями или непосредственно аппаратными средствами. Как и LDD, PDD может быть сконфигурирован для выполнения инициализации во время загрузки.

К тому времени, когда ядро готово запустить планировщик, оно требует ресурсов, которые строго не определены архитектурой центрального процессора, которые обеспечиваются расширениями. Эти расширения являются специфичными для конкретной платформы, на которой запускается ОС Symbian, и разрешаются для портирования изготовителем телефона, без непосредственной перекомпиляции ядра ОС.

Расширение состоит из двух компонентов: слой, независимый от платформы (PIL), и слой, определяемый платформой (PSL). Перечисленные слои аналогичны слоям LDD/PDD у драйверов устройств. PIL в основном отвечает за обеспечение функциональных возможностей, одинаковых у версий расширения, и определяет API с PSL, который обеспечивает связь непосредственно с аппаратными средствами.

ЛИТЕРАТУРА

1. Танненбаум Э. Современные операционные системы. СПб: Питер, 2002. - 1040 стр.

2. Aloni D. Cooperative Linux. // Proceedings of the Linux Symposium, vol. 1. Ottawa, Canada, 2004. - p. 3 – 11.

3. Столлингс В. Операционные системы. - Мн.: Вильямс, 2004. - 848 стр.

4. Symbian OS system definition - detailed view. Symbian develoler network. http://developer.symbian.com/main/downloads/papers/SymbOS_cat/ SymbianOS_cat.html. 2007.

5. Горнаков С. Symbian OS. Программирование мобильных телефонов на C++ и Java 2 ME. - Мн.: ДМК, 2005. - 448 стр.

6. Brash D. The ARM Architecture Version 6 (ARMv6). ARM White Paper, January 2002. - 15 pp.

УДК 004.514.62

Калиновский Р.В.

Научный руководитель: к.т.н. Костюк Д.А.

ТЕХНОЛОГИЯ ВИРТУАЛИЗАЦИИ ДЛЯ СМАРТФОНОВ

В настоящее время наблюдается бурное развитие технологий виртуализации операционных систем, позволяющих получить все многообразие программных платформ пользователя на единой аппаратной платформе вне зависимости от ее типа. Всплеск популярности вызван в первую очередь возросшими вычислительными возможностями аппаратуры, а также наличием значительного количества системного программного обеспечения с открытым исходным кодом, что облегчает его изучение и модификацию.

Современные многофункциональные устройства мобильной связи, известные также как смартфоны, при портативных размерах имеют аппаратные и программные ресурсы, зачастую сравнимые с таковыми у персональных компьютеров начального уровня (а по

отдельным показателям и превышающие их). Однако средства виртуализации на данной платформе носят зачаточный характер, ограничиваясь эмуляторами различных бытовых игровых компьютеров. Вместе с тем, из-за многообразия и экзотичности программных платформ смартфонов, рынок прикладного программного обеспечения для них далек от насыщения. С этой точки зрения виртуализация полнофункциональной прикладной платформы представляется актуальной, т.к. обеспечивает простую возможность переноса соответствующего ей парка программного обеспечения.

Использование виртуализации в вычислительных системах может быть продиктовано различными причинами. Наиболее распространенная цель ее применения на серверах - консолидация сервера - обосновывается тем, что при виртуализации нескольких загруженных не до конца систем на одном сервере возможен выигрыш в мощности, пространстве, охлаждении и администрировании из-за наличия меньшего количества физических серверов. Поскольку определить степень использования сервера может оказаться не так просто, возможна поддержка технологий виртуализации, называемой оперативным перемещением или горячей миграцией. Технология оперативного перемещения позволяет переносить на новый сервер операционную систему и его приложения, чтобы балансировать загрузку по доступным аппаратным средствам.

Также виртуализация активно применяется при разработке программного обеспечения. Отказ ядра операционной системы (ОС) или любого из его драйверов (в случае монолитных ядер) приводит к краху всей ОС. Виртуализация позволяет одновременно выполнять несколько ОС, и, если одна из них терпит крах из-за ошибки, другие ОС продолжают работать. Это упрощает отладку ядра до уровня отладки в пространстве пользователя.

К этой же категории относится использование средств виртуализации для исследования потенциально-опасного программного кода (вирусные и шпионские программы).

Последней областью применения виртуализации является запуск инородного прикладного программного обеспечения. В той или иной степени эмуляция частей операционных систем использовалась довольно давно. Так, для запуска пользовательских приложений Windows 3.1 в операционных системах Apple MacOS и в IBM OS/2 использовалось специальное программное обеспечение, имитировавшее прикладные программные интерфейсы Windows. Схожую функцию выполняет проект Wine для ОС Unix.

Другим примером виртуализации прикладного уровня является технология Rosetta, применяемая в последних версиях Apple MacOS X в связи с переходом с аппаратной платформы Power PC на платформу Intel. С помощью данной технологии пользователь получает возможность прозрачно запускать на современных компьютерах программное обеспечение, написанное под прежнюю аппаратную платформу. Данный метод несет значительную вычислительную нагрузку. Однако данные издержки более чем оправданы, учитывая обширный парк прикладного программного обеспечения, накопленный для платформы Power PC и благодаря эмуляции аппаратных средств доступный на современных компьютерах Apple.

Три названные категории задач, решаемых в с помощью виртуализации - консолидация сервера, виртуализация для исследования и отладки, виртуализация для переноса программного обеспечения – в большей или меньшей степени актуальны для смартфона. Наиболее целесообразной представляется все же виртуализация для запуска инородного прикладного программного обеспечения. Само по себе это является наиболее быстрым способом перенести накопленную базу приложений на новую платформу. Однако, если персональная ЭВМ допускает установку нескольких операционных систем с

последующей их поочередной загрузкой для решения различных задач, то в случае устройств мобильной связи это невозможно. Типичная операционная система смартфона выполняет также основные функции коммуникаций. Установка на устройство более универсальной ОС приводит к тому, что основное назначение – связь – становится недоступным. Таким образом, использование виртуализации и гостевых ОС является единственным приемлемым способом расширения базы прикладных приложений смартфона за счет быстрого переноса кода из операционных систем общего назначения.

Существуют различные способы реализовать виртуализацию. Наиболее распространенные методы - аппаратная виртуализация, полная виртуализация и паравиртуализация.

Аппаратная эмуляция, самый сложный из методов виртуализации, предполагает, что на основной системе создаётся аппаратная виртуальная машина (ВМ), эмулирующая необходимые аппаратные средства. Главная проблема аппаратной эмуляции - высокие требования к вычислительным ресурсам. Поскольку каждая команда должна моделироваться на используемом оборудовании, происходит замедление выполнения в сотни раз. Если эмулируемое аппаратное обеспечение включают конвейеры центрального процессора и блоки кэширования ветвлений, фактическое уменьшение скорости может быть порядка тысяч раз. Аппаратная эмуляция имеет и преимущества. Например, использование аппаратной эмуляции позволяет выполнить немодифицированную операционную систему, предназначенную для другой аппаратной платформы. Одно из наиболее востребованных применений аппаратной эмуляции - одновременная разработка встроенного программного обеспечения и аппаратных средств.

Модификацией аппаратной эмуляции является метод двоичной трансляции, при которой инструкции эмулируемой аппаратной платформы налету переводятся в команды платформы реальной. Такой подход позволяет несколько увеличить скорость выполнения программ приложений. Наибольшее распространение этот метод получил однако в сфере кроссплатформенных систем программирования, таких как Java и NET, в которых кроссплатформенность достигается тем, что программы транслируются в платформонезависимый байт-код и выполняющихся на виртуальном оборудовании, не соответствующем в действительности ни одной из существующих аппаратных платформ. Трансляция одного участка кода производится один раз за время работы программы, что и обеспечивает прирост в производительности. Хотя использование двоичной трансляции дает преимущества, вместе с тем усложняется работа виртуальной машины и налагаются на нее дополнительные требования.

Полная виртуализация является другим распространенным методом. Эта модель использует монитор виртуальной машины, который является посредником между гостевыми операционными системами и аппаратными средствами. Определенные защищенные инструкции должны быть перехвачены и обработаны гипервизором, который обеспечивает разделение реального оборудования между операционными системами. Полная виртуализация быстрее, чем аппаратная эмуляция, но из-за посредничества гипервизора выполнение программ все-таки замедленно. Огромное преимущество полной виртуализации - отсутствие необходимости модифицировать операционную систему. Единственное ограничение состоит в том, что операционная система должна поддержать используемое оборудование.

Паравиртуализация имеет некоторое сходство с полной виртуализацией. Этот метод использует гипервизор для распределенного доступа к используемому оборудованию, но интегрирует выполняющий виртуализацию код в операционную систему непосредственно. Этот подход устраняет потребность в любой перекомпиляции или перехвате, потому что сами операционные системы сотрудничают в процессе виртуализации.

Недостатком паравиртуализации является то, что при реализации метода необходимо модифицировать гостевые операционные системы, однако достигаемая скорость выполнения не намного ниже, чем системы без виртуализации. Как и у метода полной виртуализации, возможна поддержка нескольких различных операционных систем одновременно.

Современный смартфон является полнофункциональным широкопрофильным компьютером, возможности которого далеко выходят за пределы изначальной коммуникационной функции, охватывая ведение баз данных, получение и обработку изображений, работу с файловой системой и файлами различных форматов на сменных носителях, выполнение игровых и развлекательных задач. Средства виртуализации могли бы еще более расширить возможности таких устройств.

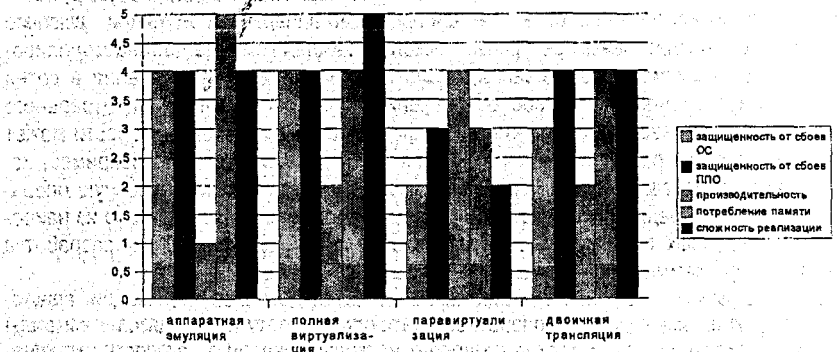

Рисунок 1 – Сравнение технологий виртуализации

На рис. 1. приведен результат сравнения описанных технологий виртуализации, приведенный к условной пятибалльной шкале. В число сравниваемых характеристик были включены способность ОС и прикладного программного обеспечения противостоять сбоям в работе системы, производительность, потребление памяти и сложность реализации. Смартфон накладывает на них следующие требования:

- Способность ОС и прикладного программного обеспечения противостоять сбоям в работе системы – умеренно-высокая (поскольку смартфон обычно не относится к числу критичных к случайным сбоям устройств, таких как системы жизнеобеспечения, медицинского и технического мониторинга и др.).
- Производительность – высокая, учитывая ограниченные ресурсы устройства.
- Потребление памяти – низкое, также исходя из ограниченных аппаратных ресурсов.
- Сложность реализации – низкая.

Как видно из рисунка, технология паравиртуализации идеально соответствует выбранным критериям.

ЛИТЕРАТУРА

1. Танненбаум Э. Современные операционные системы. - СПб: Питер, 2002. - 1040 стр.
2. Aloni D. Cooperative Linux. // Proceedings of the Linux Symposium, vol. 1. Ottawa, Canada 2004. - p. 3 – 11