

Виртуализация в Linux

Кирилл Колышкин <kir@openvz.org>

Кирилл Коротаев <dev@openvz.org>

24 августа 2006 г.

Аннотация

В докладе даётся обзор основных типов виртуализации (эмуляция, паравиртуализация, виртуализация на уровне ОС), даются примеры реализации всех трёх технологий, их сравнение и области применения. Подробно рассматривается виртуализация на уровне ОС на примере OpenVZ. Описываются основные компоненты ядра (изоляция и виртуализация, управление ресурсами, чекпойнтинг и миграция) и пользовательских утилит. Кратко демонстрируются основные возможности OpenVZ (на реальной системе).

1 Виртуализация. Типы виртуализации

В контексте данного доклада *виртуализация* — это система или методология разделения ресурсов компьютера на множество независимых сред. Возможно выделить четыре типа виртуализации: эмуляция, паравиртуализация, виртуализация на уровне операционной системы и многосерверная (кластерная) виртуализация (последняя в данном докладе не рассматривается).

Каждый из этих типов имеет свои достоинства и недостатки, обуславливающие сферы применения решений, на нём основанных.

Эмуляция позволяет запускать произвольную операционную систему без необходимости её изменения, но обладает сравнительно низкой производительностью, масштабируемостью и плотностью размещения. Примеры реализации: продукты VMware, QEmu, Bochs, Parallels.

Паравиртуализация даёт лучшую производительность, чем эмуляция, но требует изменения «гостевой» операционной системы. Примеры реализации: Xen, UML.

Виртуализация на уровне операционной системы обладает наилучшей возможной производительностью и плотностью, а также наиболее полно реализует динамическое управление ресурсами. В то же время эта технология не позволяет одновременно запускать несколько ядер разных ОС на одном сервере. Примеры реализации: FreeBSD Jail, Solaris Zones/Containers, Linux-VServer, OpenVZ.

2 Понятие VE

Виртуальная среда (VE, также используются термины VPS, VDS, контейнер (container), раздел (partition) и т.п.) — это отдельная изолированная среда выполнения программ, которая (с точки зрения её владельца и запущенных в VE программ) выглядит как обычный выделенный сервер.

VE имеет свои процессы (начиная с `init`), файловую систему, пользователей (включая `root`), сетевое устройство с IP-адресами, таблицы маршрутизации, правила сетевого фильтра (`netfilter/iptables`) и так далее.

Множество VE сосуществует в рамках одного физического сервера. В разных VE могут работать разные дистрибутивы Линукса, однако все VE работают под одним и тем же ядром.

3 Ядро OpenVZ

Ядро OpenVZ — это модифицированное ядро Linux, которое добавляет следующую функциональность: виртуализация и изоляция отдельных подсистем ядра, управление ресурсами, а также чекпойнтинг.

Виртуализация и изоляция позволяют в рамках единого ядра ОС создавать множество виртуальных сред, каждая из которых имеет свои собственные процессы, файлы, сетевые и другие устройства и т. п. *Подсистема управления ресурсами* позволяет ограничивать и делить между VE такие ресурсы, как процессорное время, память и дисковое пространство. *Чекпойнтинг* — это процесс «замораживания» VE с сохранением полного образа её состояния в файл и возможностью последующего полного восстановления рабочего состояния VE.

Система управления ресурсами в OpenVZ состоит из трёх компонентов:

1. *Двухуровневая дисковая квота.* Администратор сервера OpenVZ может установить дисковые квоты на VE в терминах дискового

пространства и количества айнодов (i-nodes). Это первый уровень дисковой квоты.

В дополнение к этому администратор VE (`root`) может использовать обычные утилиты внутри своей VE для настроек стандартных дисковых квот UNIX для пользователей и групп.

2. *«Честный» планировщик процессов.* Планировщик процессора в OpenVZ также двухуровневый. На первом уровне планировщик решает, какой VE дать квант процессорного времени, базируясь на значении параметра `cpuunits` для VE. На втором уровне стандартный планировщик Linux решает, какому процессу в выбранном VE дать квант процессорного времени, базируясь на стандартных приоритетах процесса в Линуксе и т. п.
3. *User Beancounters.* Это набор счётчиков, ограничений и гарантий для различных ресурсов на каждое VE. Контролируется примерно 20 параметров, которые тщательно выбраны для того, чтобы никакая VE не могла злоупотребить каким-либо ресурсом, который ограничен для всего сервера, и, таким образом, помешать другим VE.

Ресурсы, которые считаются и контролируются — это в основном оперативная память и различные объекты в ядре, например, разделяемые сегменты памяти IPC, сетевые буферы и т. п.

Чекпойнтинг позволяет производить «живую» миграцию VE на другой физический сервер. VE «замораживается» и её полное состояние сохраняется в файл на диске. Далее этот файл можно перенести на другую машину и там «разморозить» (восстановить рабочее состояние) VE. Задержка этого процесса (время, когда VE в «замороженном» состоянии) составляет примерно несколько секунд. Важно подчеркнуть, что это задержка в обслуживании, но не отказ в обслуживании, так как разрыва сетевых соединений не происходит.

4 Утилиты OpenVZ

В состав OpenVZ входит набор утилит `vzctl`, предоставляющий довольно высокоуровневый интерфейс командной строки для создания и управления виртуальными средами. Так, например, для создания и запуска новой VE необходимо всего две команды — `vzctl create` и `vzctl start`.

Для изменения различных параметров VE служит команда `vzctl set`. Все ресурсы (например, размер виртуальной памяти) могут быть изменены во время работы VE. В других технологиях виртуализации (напр. эмуляции или паравиртуализации) это либо достаточно сложно, либо невозможно.

Существует также набор утилит `vzpkg`, реализующий функциональность создания и обновления образов VE для различных дистрибутивов. В текущей реализации `vzpkg` базируется на утилите `yum`, однако в будущем планируется добавить поддержку `apt`.

5 Области применения

Средства виртуализации на уровне ОС дают возможность воспользоваться всеми преимуществами виртуализации, не платя при этом большую цену в виде падения производительности.

- *Консолидация серверов* позволяет сэкономить на стоимости оборудования, площади серверных площадок и затратах на обслуживание.
- Весьма привлекательно использование технологий виртуализации на уровне ОС для *хостинга* ввиду низкой себестоимости VE, возможностей массового управления, быстрого создания VE. Однако следует заметить, что в этом сценарии надёжная изоляция и управление ресурсами приобретают критическую важность.
- *Безопасность* может быть существенно улучшена при разнесении разных сетевых сервисов (демонов) в разные VE. В качестве дополнительных преимуществ пользователь получает динамическое управление ресурсами и возможность «живой» миграции.
- Использование VE в *процессе разработки и тестирования программного обеспечения* выглядит привлекательно ввиду хорошей производительности, возможностей иметь большое количество VE, различные дистрибутивы, версии библиотек, производить клонирование VE и т.п.
- В *образовательных целях* VE можно использовать, чтобы дать каждому студенту права `root`, не требуя для этого большого количества серверов. Возможность пробовать разные дистрибутивы, а также быстро создавать новые VE сводит к минимуму необходимость в частой переинсталляции.