с некоторыми новыми средствами, более удобными в том плане, что они напоминают скорее браузер, а не просто робот. Но проблем по-прежнему полным-полно, и команда Archive It проводит кучу времени в поиске новых способов сбора информации и просмотра различных видов медиа. Это удобнее делать на менее масштабных проектах Archive It, поскольку там они могут плотнее взаимодействовать с клиентами, ведь в таких крупных подборках, с которыми работает моя команда, в миллиард страниц за неделю, гораздо сложнее понять, что ты что-то упустил.

В итоге, приходится много анализировать, чтобы понять, где есть пробелы. Изменения необходимы, чтобы маленькие победы сменялись большими. Архивировать Интернет становится все труднее.

LXF: Какое открытое ПО вы используете?

AP: Ну, Wayback Machine сама по себе открытый код, как и наши программы для чтения книг, да и весь наш стек, насколько это возможно. Естественно, все это на Linux. У нас есть сайт под названием Open Library (https://openlibrary.org), где содержится миллионный каталог книг, доступных для скачивания (если позволяет авторское право) или просмотра. Все это Open Source, и люди вносят сюда свой вклад. У нас был опыт открытого проекта, и в основном все изменения делали мы. Естественно, какие-то вещи мы выгоды ради не открываем. Но мы также используем *JWPlayer* для воспроизведения аудио и видео, и следим, чтобы все переводилось в открытые форматы — поэтому мы все еще переводим из MP3 в OGG, хотя многие его не используют.

Мы пытались поработать с Wikipedia — у них есть Wikimedia Commons, где хранится много медиа-записей, но им надо, чтобы это все содержалось в открытых форматах. А люди не умеют создавать файлы OGG, и в итоге Wikimedia много чего недополучает. И мы затеяли небольшой эксперимент с Wikimedia в Германии, разрешив людям загружать все, что угодно, хоть в MP4, хоть в WMV, а мы переводили это в OGG, чтобы Wikipedia забирала это и размещала в Wikimedia Commons. Я считаю, что такое переформатирование очень важно, даже при том, что открытые форматы на нашем сайте не пользуются самым большим спросом.

О ТОМ, КУДА ИДУТ ДЕНЬГИ

Вся работа, которую мы делаем за деньги... на благо общества.

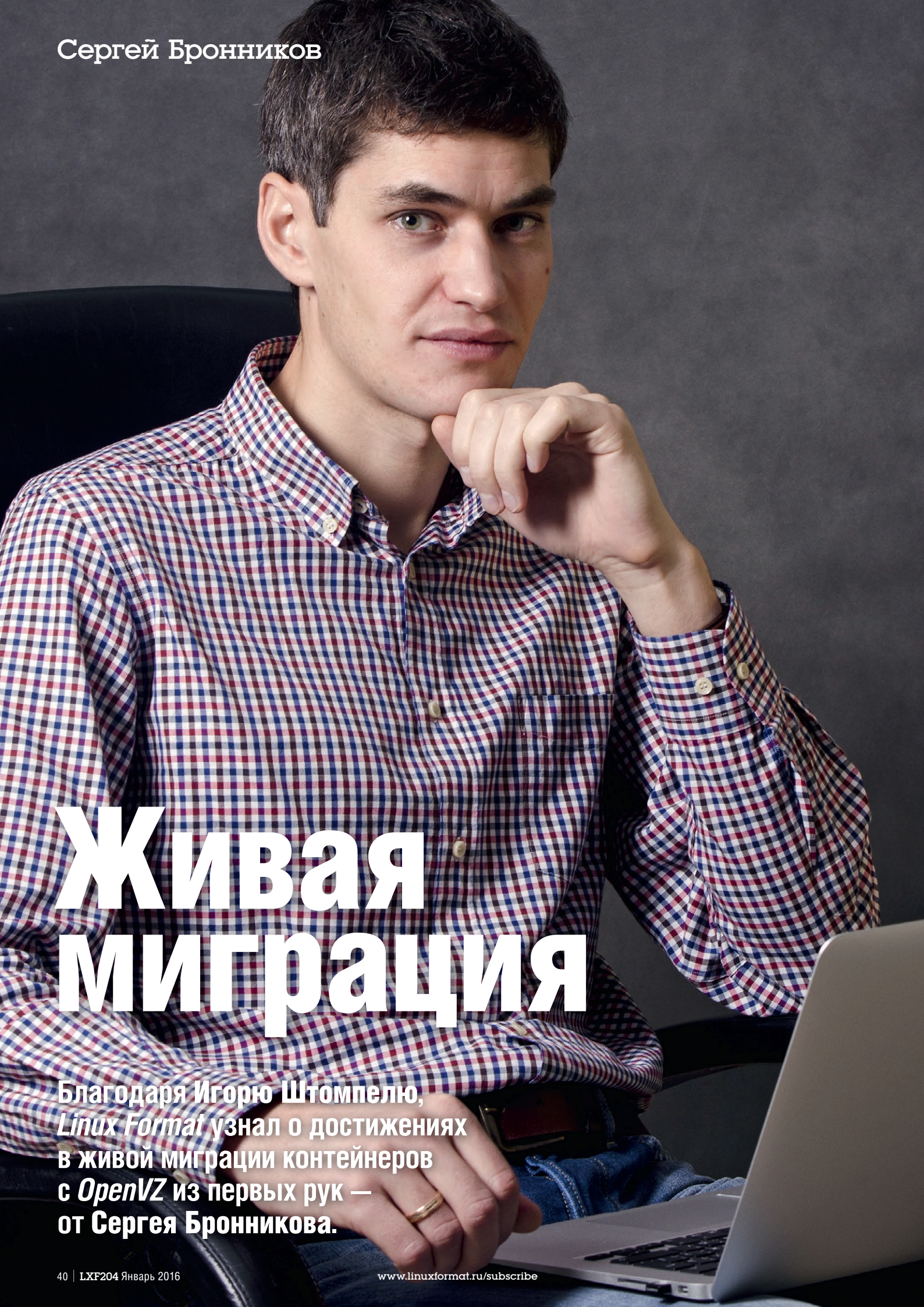
LXF: Расскажите о вашем докладе на OSCON в этом году.

AP: Мы подготовили его совместно с Вики Брассер [Vicky Brasseur], которая раньше работала в Archive. Она живет здесь, в Портленде, и теперь является частным консультантом, но по образованию она инженер. Так вот я говорила об Internet Archive в целом, а она рассказывала об API, с помощью которых можно закачивать и выкачивать материалы. Мне кажется, это очень ценно для людей, организация у нас дружная, но не идеальная. У нас не самая лучшая документация, и 15 человек не дожидаются вашего запроса, готовые ответить. И я думаю, полезно выступать на таких мероприятиях, как OSCON, поскольку это дает людям шанс узнать, чем мы располагаем, и понять, что у нас можно запрашивать. Мы хотим помогать людям, а значит, хотим чтобы Archive пополнялся и был востребован.

LXF: На OSCON царит атмосфера разнообразия и дружелюбия, но каков ваш личный опыт в том, что значит быть женщиной в сфере технологии?

AP: Боже мой! Ну, сейчас все куда лучше. В 1996-м, когда я начинала, все было довольно, м-м, любопытно: бывало, меня просили покинуть собрание со словами «Ну, дальше, дорогая, уже наши технарские дела», или «секретарша ушла на перерыв, ты ее не подменишь — у нас же больше женщин нет, а на звонки отвечать надо?» Теперь это прекратилось, Internet Archive — компания очень разношерстная, кого только нет. Но мужчин по-прежнему больше, такова уж специфика инженерных талантов. Как и на подобных конференциях: сегодня с утра я встретила здесь около 400 человек, из них женщин — всего с десяток. LXF

Сергей Бронников



Живая миграция

Благодаря Игорю Штомпелю, *Linux Format* узнал о достижениях в живой миграции контейнеров с *OpenVZ* из первых рук — от Сергея Бронникова.

Сергей Бронников, менеджер Open-Source-проекта *OpenVZ*, занимается развитием *OpenVZ* и взаимодействием с сообществом. Ранее в компании Parallels как SQA-менеджер [SQA — Software Quality Assurance, контроль качества ПО] занимался тестированием облачного сервера *Virtuozzo* и *Virtuozzo for Windows* (ранее — *Parallels Cloud Server* и *Parallels Containers for Windows*).

Linux Format: Расскажите, чем вы занимаетесь в Open Source.

Сергей Бронников: Процесс разработки наших продуктов всегда тесно переплетался с открытыми проектами: это и непрекращающийся процесс интеграции патчей ядра *OpenVZ* в основную ветку ядра Linux, и публикация исходного кода компонентов *Virtuozzo* под свободной лицензией. А в последнее время мы стали более активным участником мира Open Source за счет участия в сторонних проектах.

Все проекты с открытым исходном кодом, в которых мы участвуем, можно поделить на две категории:

» Проекты, которые изначально были организованы сотрудниками компании, и результат этих проектов является частью наших коммерческих продуктов. Примеры таких проектов — уже успевший стать известным CRIU, проект P.Haul, библиотека для управления контейнерами *LibCT*, проект по управлению памятью в контейнерах *vcmmd*.

» Проекты, в которых мы участвуем на правах контрибьюторов: интеграция *Virtuozzo* с компонентами *OpenStack* (Nova, Cinder), разработка драйвера для *Virtuozzo* в проекте LibVirt, разработка недостающей функциональности в QEMU, участие в разработке библиотеки *runc* для запуска контейнеров.

Это все, что касается участия в разработке открытых проектов. А если говорить более глобально, то мы свою цель сформулировали давно: сделать наши контейнеры самыми популярными. И эта цель поставлена в виде ошибки номер один — OVZ-1 — в нашем публичном баг-трекере. Закроем ее как исправленную, когда поймем, что достигли цели.

LXF: Сколько активных разработчиков на сегодняшний день?

СБ: По понятным причинам, наиболее активными разработчиками компонентов *Virtuozzo*/*OpenVZ* являются сотрудники компании. Около 30 моих коллег заняты выпуском обновлений для стабильных версий продуктов и разработкой новых версий. Помимо штатных сотрудников, в разработке компонентов *Virtuozzo* участвуют инженеры, которые никак не связаны с компанией, и их количество всегда зависит от разных факторов: популярность и востребованность проекта, его зрелость, уровень знаний, необходимый для участия в проекте, и т.д.

Компонент может являться востребованным, но количество его активных разработчиков — малым. Как, например, с утилитой для управления контейнерами *vzctl*: активный разработчик только один — Кирилл Колышкин, а в разное время заплатки-патчи [patch] для *vzctl* присылали более 200 человек. То есть людей зачастую устраивала существующая функциональность, и они только исправляли баги или добавляли нужные им опции.

Миграция Linux-контейнеров — не единственный сценарий использования CRIU, технология находит применение и в других областях. Поэтому в разработке успели поучаствовать около 60 человек, хотя наиболее активными участниками являются около 13. Чтобы не утомлять вас сухими цифрами, советую посмотреть количество контрибьюторов для каждого компонента *Virtuozzo* в учетной записи *OpenVZ* на GitHub.

LXF: В августе 2015 года на конференции Linux Plumbers Павел Емельянов представил P.Haul. В чем его особенности и назначение? Какое место эта технология занимает среди *OpenVZ* и CRIU?

СБ: Технология P.Haul позволяет выполнять живую миграцию контейнеров с одного физического сервера на другой. С точки зрения реализации код живой миграции контейнеров всегда был сосредоточен в ядре Linux, а со стороны пользователя была утилита, которая только управляла миграцией. То есть все «волшебство» по замораживанию контейнеров происходило в ядре. Мы предпринимали попытки интегрировать свои патчи для «заморозки» и «разморозки» контейнеров в основную ветку ядра Linux, но по разным причинам этого сделать не получалось. Впрочем, этого не получилось и у других проектов, которые также были заинтересованы в добавлении функциональности сохранения и восстановления состояния процессов в «ванильном» ядре. Поэтому мы решили сделать ход конем: вынести большую часть кода из ядра Linux в пространство пользователя, а ядро использовать только как источник информации о процессах. У нас это получилось, и родился проект CRIU (Checkpoint and Restore In Userspace). Утилита, разрабатываемая в рамках этого проекта, позволяет «заморозить» процесс (или группу процессов), сохранить состояние в файлах на диске и потом восстановить состояние этих процессов из файлов. Процесс продолжит работать как ни в чем не бывало. Но сохранение и восстановление процессов — это только часть технологии «живой» миграции контейнеров. Основная сложность технологии в том, что нельзя просто так взять и «заморозить» процессы и начать переносить состояние этих процессов на другой сервер. Потому что это будет долго, и «живой» миграции не получится. Вместо этого надо либо сначала перенести память процессов, а потом «заморозить» их и перенести всё остальное; или сначала всё перенести, а память подключить как swap, чтобы она переносилась на другой сервер по мере востребованности процессами. На этом этапе и нужен P.Haul, который управляет процессом миграции. Дополнительно он может заниматься переносом файловой системы контейнера на другой сервер, если не используется разделяемое хранилище данных. Собственно, название проекта и отражает его назначение — Process Hauler, «перевозчик процессов».

О ПРОЕКТЕ CRIU

Решили вынести часть кода из ядра Linux в пространство пользователя.

LXF: Реализована ли поддержка P.Haul за пределами *OpenVZ* — например, в *LXC* или других проектах?

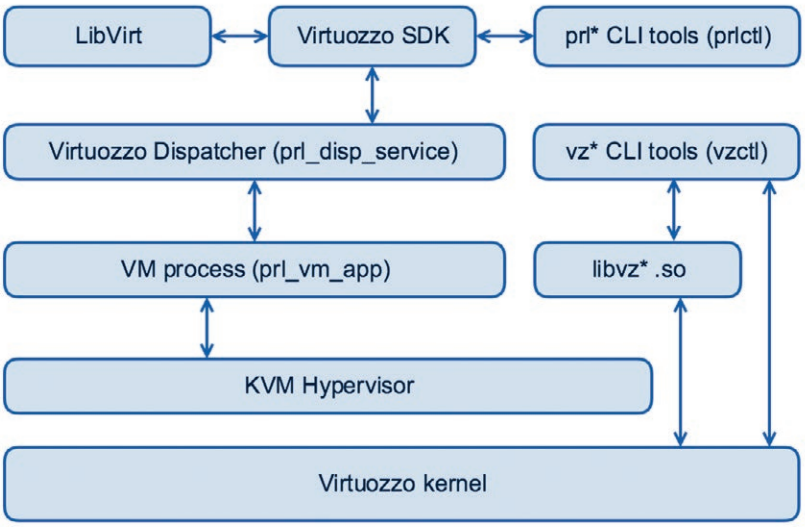
СБ: *OpenVZ* — не единственная реализация технологии контейнеров для Linux, поэтому при разработке новых компонентов мы стараемся их проектировать так, чтобы была возможность их использования в других проектах. Например, возможность заморозки и восстановления контейнеров с помощью CRIU была добавлена в *LXC* примерно год назад разработчиками Canonical при участии разработчиков CRIU. Разработка функциональности C/R для Docker продолжается, и есть надежда, что в этом году она появится в основной ветке репозитория. С проектом P.Haul похожая история, но относительно проектов все с точностью до наоборот. Патчи для интеграции P.Haul с Docker были приняты разработчиками P.Haul примерно месяц назад. А интеграция с *LXC* еще впереди: разработчики *LXC* выразили заинтересованность в интеграции с P.Haul и обещали сделать ее в ближайшее время. В *Virtuozzo 7* живая миграция будет реализована с помощью CRIU/P.Haul уже в Beta 3.

LXF: Расскажите о портировании патчей в ядро RHEL. Как организован процесс, в чем его особенности? Как организовано взаимодействие проекта *OpenVZ* и Red Hat?

СБ: Чтобы полнее описать процесс разработки, нужно сделать небольшой экскурс в историю разработки *OpenVZ*. На ранних этапах разработки контейнеров *Virtuozzo* мы использовали «ванильные» Linux-ядра. То есть на новый релиз ядра от Линуса Торвальдса мы накладывали наши патчи с реализацией контейнеров, тестировали своими силами и выпускали новое ядро *OpenVZ*. Но в 2006 году мы переключились на использование ядер из дистрибутива Red Hat Enterprise Linux и новое ядро стали делать на основе ядра из RHEL 4. Red Hat Enterprise Linux — это дистрибутив для предприятий от компании Red Hat, компании, чье имя всегда стоит на самом верху списка тех компаний, которые внесли самый большой вклад в разработку ядра Linux.

Конечно, никакое ядро не может быть идеальным и безошибочным, но ядро из их дистрибутива — довольно хороший пример качественного и стабильного ядра.

Разработка ядра для RHEL выглядит так: инженеры из Red Hat создают ветку последней версии «ванильного» ядра, для которой объявляется долгосрочная поддержка (такие ветки поддерживает Грег Кроа-Хартман), тестируют его, попутно »



› Рис. 1. Архитектура первой версии *Virtuozzo*.

исправляя найденные ошибки, и портируют новые драйвера и исправления проблем в безопасности из основной ветки. Такая «подготовка» ядра к релизу продолжается примерно полгода или немногим больше. Так что к моменту релиза это ядро уже «древнее и устаревшее» — во всяком случае, так кажется, если смотреть на его версию. На самом деле никакое оно не древнее и не устаревшее, а просто более стабильное и безопасное. После релиза это ядро очень хорошо поддерживается — добавляется поддержка современного «железа», оперативно исправляются проблемы в безопасности. И все это на протяжении нескольких лет.

После релиза новой версии дистрибутива RHEL мы берем исходный код Linux-ядра этого дистрибутива и кропотливо начинаем переносить все наши патчи с предыдущей версии на новое ядро. Мы этот процесс называем «ребейзом [от *англ.* rebase]» на новое ядро. Этот процесс не всегда прост, потому что мажорные версии ядер содержат очень много изменений в разных подсистемах ядра и часть патчей приходится переделывать, а часть из них не появляется в новом ядре из-за того, что нужная функциональность появилась в основной ветке Linux-ядра. Так получилось с «живой» миграцией, поддержкой NFS в контейнерах, пространствами имен IPC, MNT, UTS, PID и NET, технологией управления памятью в контейнерах, то есть вся та функциональность, которую мы ранее «продали» в upstream [в «вышележащее» ПО, основную ветку]. До 2015 года мы весь исходный код ядер *OpenVZ* выкладывали на сайте в виде архивов; из-за этого следить за процессом разработки и сравнивать исходный код разных версий было неудобно. Поэтому разработку на базе ядра RHEL7 мы ведем в публичном репозитории, и все патчи проходят через почтовый список рассылки devel@openvz.org. В репозитории ядра все «переезды» на новую версию ядра RHEL отмечаются тэгами: 19 февраля 2015 года мы начали использовать ядро RHEL7 kernel-3.10.0-123.1.2.el7, а текущая версия нашего ядра базируется на версии rh7-3.10.0-229.7.2. В разработке ядра *OpenVZ* мы активно используем ветки, это позволяет нам разделять функциональность разной степени готовности и реже ломать сборки продукта.

Нужно отметить, что сотрудничество разработчиков *OpenVZ* и Red Hat взаимовыгодное: мы используем их качественные ядра, но сообщаем обо всех найденных проблемах во время тестирования. В последних анонсах Red Hat отметил Владимира Давыдова за обнаружение серьезных уязвимостей CVE-2014-0203 и CVE-2014-4483 в последнем обновлении ядра RHEL6 (вторая проблема, кстати, была найдена с помощью одного из наших автоматических тестов, использующих Linux Test Project). Василий Аверин получил благодарность за обнаружение ошибки CVE-2014-5045, а Дмитрий Монахов — за CVE-2012-4508 и CVE-2015-8324.

LXF: Расскажите об архитектуре проекта *OpenVZ*. В чем ее особенности? Как устроен *OpenVZ* изнутри, что представляет собой на уровне исходного кода — патчи к ядру или комбинацию патчей и другого, а также на уровне исполнения — в чем выражается его работа как программы?

СБ: Процесс разработки я подробно описал в ответе на предыдущий вопрос.

Архитектура *OpenVZ* стабильной версии не представляет собой ничего интересного и сводится к взаимодействию двух компонентов: ядра Linux и пользовательской утилиты *vzctl*, которая управляет контейнером с помощью системных вызовов к ядру. Архитектура новой версии *Virtuozzo* гораздо интереснее. Но прежде чем рассказывать про компоненты *Virtuozzo*, я сделаю небольшой экскурс в историю разработки *Virtuozzo*. Первая коммерческая версия продукта с Linux-контейнерами имела одинаковую с *OpenVZ* архитектуру: ядро Linux и пользовательские утилиты. А первая версия *Virtuozzo* (она же *Parallels Cloud Server* и *Parallels Server Bare Metal*), которая объединила два разных типа виртуализации, уже имела более продвинутую архитектуру, на базе клиент–серверной модели: диспетчер, управляющий виртуальными машинами и контейнерами, и интерфейс клиента в виде GUI и консольных утилит, работающих через SDK *Virtuozzo* (как показано на рис. 1).

Такая архитектура позволила предоставлять API для интеграции с решениями других компаний. В ранних версиях *Virtuozzo* (4, 5) утилиты для работы с контейнерами и виртуальными машинами были недостаточно интегрированы; для операций с виртуальными машинами использовались универсальные идентификаторы, а для контейнеров — номера, но в *Virtuozzo 7* мы исправили эти недостатки. Для обоих типов окружений можно будет использовать одинаковые утилиты: *prlctl*, *pbackup* и т. д.

LXF: Реализацию какой функциональности можно ожидать в Virtuozzo в ближайшее время? Что в долгосрочных планах?

СБ: Переход на использование *KVM* и *cgroups*.

LXF: Частично вы об этом говорили, но хотелось бы заострить внимание. Скажите, какие ветки ядер *OpenVZ* поддерживаются на данный момент? Чем они различаются?

СБ: Так как ядро *OpenVZ*/*Virtuozzo* базируется на ядре от Red Hat, то мы различаем пять веток ядер *OpenVZ*:

» по две ветки -testing и -stable для ядер, базирующихся на RHEL5 и RHEL6. Эти ядра используются в текущей версии *OpenVZ* и коммерческих версиях *Virtuozzo* 5 и 6.

» ветка -testing с ядрами, основанными на RHEL7. Эти ядра попадают в тестовые сборки дистрибутива Virtuozzo 7, который находится в разработке.

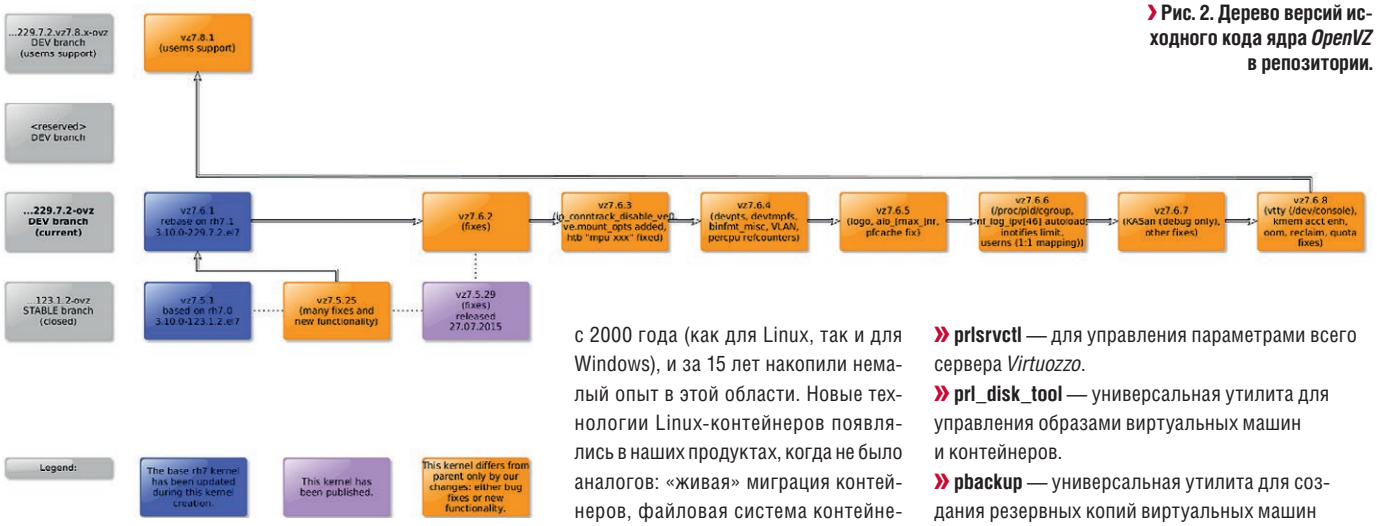
Если говорить о ветках в репозитории исходного кода ядра *OpenVZ*, то там чуть сложнее. Для каждого «ребейза» на новое ядро Red Hat, для серьезных изменений и новой функциональности мы делаем отдельную ветку в репозитории (рис. 2).

LXF: Расскажите о масштабировании в *OpenVZ*.

СБ: Контейнерная виртуализация в Linux — это комбинация технологии изоляции процессов (пространства имен) и технологии управления ресурсами (CGroups). Каждая из этих технологий не меняет архитектуру ОС. Как следствие, контейнеры *OpenVZ*/*Virtuozzo* масштабируются точно так же, как и само ядро Linux — до тысячи CPU и терабайтов памяти. В разработке *OpenVZ*/*Virtuozzo* мы исходим из практических задач, а основной сценарий использования наших контейнеров — это услуги хостинга. Хостинг-провайдер всегда заинтересован в том, чтобы разместить максимальное количество изолированных окружений на одном физическом сервере, чтобы получить максимальную прибыль. Поэтому одна из метрик качества *Virtuozzo* — это плотность размещения контейнеров. Мы измеряли плотность размещения контейнеров на серверах разных конфигураций (вплоть до 64 CPU, 256 ГБ ОЗУ) и всегда упирались в ресурсы сервера (процессор, диск или память), но не проблемы масштабирования *Virtuozzo*.

LXF: В чем отличие *OpenVZ* от других технологий виртуализации (*VMware*, *Xen* и др.)? Преимущества?

СБ: Основное отличие *Virtuozzo* от похожих решений — это возможность использования обоих



› Рис. 2. Дерево версий исходного кода ядра *OpenVZ* в репозитории.

типов виртуализации в рамках одного продукта. Больше не нужно выбирать между виртуальными машинами и контейнерами. *Virtuozzo* позволит вам использовать как безопасные Linux-контейнеры с хорошей производительностью и высокой плотностью, так и ВМ. Улучшения в интеграции коснулись и пользовательских утилит (*prlctl*, *prl_disk_tool*) и API (Virtuozzo SDK, общий драйвер LibVirt).

Примеры использования утилиты *prlctl* для создания и старта контейнера и виртуальной машины:

```
prlctl create LinuxFormat_1 --vmtype=ct; prlctl start LinuxFormat_1
prlctl create LinuxFormat_2 --vmtype=vm; prlctl start LinuxFormat_2
```

Вообще в последнее время появилось много продуктов для виртуализации. И подчас даже продвинутому пользователю трудно разобраться во всех плюсах и минусах похожих решений. Мы решили помочь пользователям и подготовили таблицу на сайте проекта, в которой сравнили функциональность коммерческих версий *Virtuozzo* (5, 6, 7), бесплатных версий (*OpenVZ* и *Virtuozzo 7*), *LXC*, *MS Hyper-V*, *RHEV* и *Citrix Xen*. Несмотря на то, что таблица содержит только утвержденную функциональность «семерки», она позволит вам понять, какие задачи вы сможете решать с помощью *Virtuozzo* и достаточно ли вам будет функциональности открытой версии. Основными отличиями между коммерческой и бесплатной версиями *Virtuozzo 7* будет дополнительная поддержка гостевых ОС семейства Windows (драйвера для гостевых ОС будут снабжены сертификатами Microsoft), web-интерфейс для управления серверами *Virtuozzo*, отказоустойчивое и распределенное хранилище Virtuozzo Storage, возможность использования кластера с High Availability и каталог образов для виртуальных машин и контейнеров.

Доказывать, что *Virtuozzo* лучше, чем имярек, дело неблагодарное, потому что лучше отталкиваться от задач, которые планируется решить с помощью виртуализации. Ситуация с контейнерами похожа на состояние дел в автопроме: кто-то из автопроизводителей проектирует свои двигатели, тестирует их, внедряет новые технологии и т.д. А кто-то покупает уже готовые технологии и занимается только производством. Мы занимаемся контейнерами с 2000 года (как для Linux, так и для Windows), и за 15 лет накопили немалый опыт в этой области. Новые технологии Linux-контейнеров появлялись в наших продуктах, когда не было аналогов: «живая» миграция контейнеров, файловая система контейнера в файле (ploop) и т.д. Пространства имен, которые являются базовыми технологиями Linux-контейнеров, появились в «ванильном» ядре благодаря работе наших инженеров. А сейчас используются в любом продукте с контейнерами. Это всё стало следствием того, что вся функциональность разрабатывалась с упором на реальную эксплуатацию решения, а не на предоставление механизмов для реализации той или иной возможности, как поступают в Linux upstream containers.

LXF: Будет ли расширяться поддержка «гостевых» ОС?

СБ: Мы регулярно добавляем поддержку новых гостевых ОС. Обычно поддержка новой ОС появляется в следующем обновлении *Virtuozzo* после выхода этой ОС, хотя тестирование новых версий гостевых ОС мы начинаем задолго до выхода финальной версии. *Virtuozzo 7* включит поддержку актуальных версий ОС семейства Windows и Linux.

LXF: Планируется ли расширение пользовательского набора утилит?

СБ: Судите сами. Для текущей версии *OpenVZ* мы поддерживаем три основные утилиты:

» **vzctl** для управления контейнерами;
» **ploop** для управления образами дисков;
» **vzstats** для сбора статистики об использовании *OpenVZ*.

В *Virtuozzo 7*, помимо ploop и vzctl появляются:

» **vzstat** — для мониторинга состояния виртуальных машин и контейнеров.
» **vztt** — для управления шаблонами контейнеров.
» **prlsrvctl** — для управления параметрами всего сервера *Virtuozzo*.
» **prl_disk_tool** — универсальная утилита для управления образами виртуальных машин и контейнеров.
» **pbackup** — универсальная утилита для создания резервных копий виртуальных машин и контейнеров.
» **prlctl** — универсальная утилита для управления контейнерами и виртуальными машинами. Эта утилита в следующей версии полностью заменит *vzctl*, поэтому мы настоятельно рекомендуем нашим пользователям использовать именно ее, а не *vzctl*.
» **prl_nettool** — для управления сетевыми настройками гостевой операционной системы и ОС внутри контейнера.
» **vcmmd** — для управления памятью контейнеров.

В будущем, по мере развития продукта, мы не исключаем появления новых утилит. Исходный код всех перечисленных выше утилит доступен в основном репозитории src.openvz.org и зеркалах на GitHub — http://github.com/OpenVZ.

LXF: Что можно сказать о развитии официально неподдерживаемых утилит?

СБ: Параллельно с развитием *OpenVZ* появилось множество утилит, которые развивались вне проекта *OpenVZ*. По большей части эти утилиты дублировали функциональность утилит из платной версии *Virtuozzo* — например, утилиты для создания резервных копий контейнеров. Так как эти утилиты развивались вне проекта и людьми, не связанными с проектом, то о планах по развитию лучше спросить авторов этих утилит. Мы в свою очередь, насколько это возможно, стараемся не ломать совместимость в новых версиях *Virtuozzo*, чтобы оставалась возможность использования этих инструментов. LXF

Пробуем тестовые версии Virtuozzo 7

Разработка *Virtuozzo 7* в самом разгаре, но так как процесс разработки открыт, то все тестовые сборки доступны для тестирования (https://download.openvz.org/virtuozzo/factory/x86_64/iso/). Наличие сервера для этого совершенно не обязательно.

Можно использовать образ виртуальной машины для *Vagrant*:

```
$ vagrant init OpenVZ/Virtuozzo-7.0
$ vagrant up --provider virtualbox
$ vagrant ssh
```

Или использовать образ для виртуальной машины в Amazon EC2: надо в AWS Marketplace (https://aws.amazon.com/marketplace) найти образ *Virtuozzo 7* и запустить экземпляр [instance], используя этот образ. Использование экземпляра с минимальными параметрами (t2.micro) будет бесплатным.