Интервью с разработчиком reiser4 Эдуардом Шишкиным

Ввиду того, что Эдуард — человек занятой, эпопея с интервью растянулась на неопределённый срок. Но, несмотря ни на что, разработчик reiser4 таки выделил время и ответил на вопросы уважаемого сообщества Хабра и ЛОРа. Что из этого вышло — читайте под катом.

— Как обстоят дела с продвижением reiser4 в ядро?

Технических препятствий для этого я уже не вижу: все проблемы из знаменитого «списка для включения» решены. Осталось только прояснить отношения с VFS, а соответствующая статья для публикации пока ещё не готова.

Вообще, продвижение reiser4 в ядро Линукс имеет сейчас низкий приоритет. Просто, потом нужно будет мгновенно реагировать на все изменения в VFS/block layer. А у меня не всегда есть такая возможность. В -mm ветке же никто от меня этого не требует. Если что-то ломается, Эндрю Мортон просто шлёт уведомление. А я, когда нахожу время, исправляю.

По поводу популярных прогнозов, что «reiser4 в ядро не включат и она умрёт», хочу сказать, что не понимаю навязчивую идею «путёвки в жизнь», якобы даваемой включением проекта в основную ветку ядра Линукс. Reiser4 — это результат 18-летних исследований в области хранения данных, не привязанный к конкретной операционной системе. Результат, над которым работало много ученых. Не включат в Линуксе — включат в другой ОС, где наши идеи покажутся интересными. На Линуксе свет клином не сошёлся…

— Есть ли смысл провести что-то вроде рекламной кампании о reiser4 для улучшения её имиджа?

Лучшая рекламная кампания — это объяснить людям, как она работает. Ибо все смотрят на её код, и никто ничего не понимает. Вот вам и Open Source. Каким образом это объяснять? Только статьями, публикуемыми в авторитетных изданиях. И уж, конечно, ни

о какой Википедии не может быть и речи. Википедия хороша для освещения творчества художников эпохи Ренессанса. А страница вашего проекта здесь рискует превратиться в отхожее место для компетиторов.

Со своей стороны я собираюсь опубликовать пару статей. Первая будет собственно про модульную архитектуру, вторая будет целиком посвящена модулю «dancing tree» (*единственному пока имеющемуся плагину интерфейса «TREE»*). Это будет очень интересно: техника, применяемая в этом модуле является одной из самых изощрённых. Далее неплохо было бы объяснить, как работает менеджер транзакций и остальные плагины, но при желании это можно понять и из кода (*чего не скажешь о дереве*).

— Что лично ты думаешь о BFS (*CPU-планировщик*), BFQ (*I/O-планировщик*)?

Если честно, я давно уже не слежу за планировщиками: на своём ноутбуке кроме текстового редактора и браузера мало что запускаю. Только помню, что появлению BFS предшествовала довольно неприятная история (*кстати, характерная для модели разработки ядра Линукс*). А элеваторами Ханс раньше очень интересовался, постоянно поручал реализовывать разные свои идеи. Я лет десять назад тоже модифицировал какой-то элеватор по его заданию. Правда, лучше он от этого работать не стал. Может быть, потому, что меня не интересовали элеваторы...

— Как ты думаешь, почему более сырая и ненадёжная ФС ext4 была практически сразу принята в ядро?

Ну, же это логическое продолжение de facto стандартной файловой системы Linux ext3. Было бы удивительно, если бы им тут был красный свет.

— Как ты относишься к тому огромному количеству ФС в ядре? Оправдано ли это?

Разумеется, не оправдано. В большой степени такому зоопарку способствует устаревшая концепция VFS, которая рассматривает файловую систему как непрозрачный монолитный модуль. Раньше просто не было такого количества ФС. А сейчас разве что только ленивый не поймёт, что многие из них делают то же самое. Давно пора извлечь какие-то выводы. У

меня есть ряд предложений по улучшению ситуации (*всё будет в статье*).

— На каких выставках и форумах ты собираешься выступить в этом и следующем году?

Пока никуда не приглашали. Сам я инициативы никогда не проявляю.

— Что тебя мотивирует разрабатывать reiser4? Ведь есть множество других ФС.

Одной сильной мотивации нет. Сначала хотелось, наконец, доделать прозрачную компрессию. Потом после её анонса в 2007 году возился с Reiser4 из-за нечего делать: долго не мог найти работу. Сейчас продолжаю интересоваться некоторыми аспектами той науки хранения данных, на которой основана Reiser4. Остальные локальные ФС мне не интересны.

— Продолжаешь ли ты неформально, «за кулисами», общаться с Гансом? Принимает ли он какое-то участие в разработке?

По-другому общаться уже не получается. Он полностью отошел от computer science, хотя и пытается оказывать посильную помощь, но для полноценного участия нужен компьютер, который ему иметь не положено. А так как Ханс не может сидеть без дела, то он обложился книжками и принялся за своё старое увлечение — физику. Вот, нашел какие-то нестыковки в специальной теории относительности, просит найти российского ученого, который бы отрецензировал его новую статью. Клянёт Америку, «страну адвокатов, где науку ни во что не ставят». О России вспоминает с теплотой. С интересом следит за инициативами министра Андрея Фурсенко, который по его мнению пытается возродить былой престиж советской науки и образования. Верит, что в его проекте найдется место и иностранцам, и говорит, что вообще готов перебраться в Россию и выучить, наконец, русский язык.

— Какая у тебя основная работа?

Я работаю в отделе файловых систем компании Red Hat.

— Хватает ли времени заниматься reiser4?

Условно говоря, хватает, но только на поддержку: я обычно в курсе всех изменений, проводимых в VFS, block layer. Чтобы приспособить к ним reiser4, выходных дней обычно бывает достаточно. Разработка же означает программирование новых плагинов. Это предполагает полную занятость, не меньше. Т.е. такое возможно только за зарплату, а платить за это пока никто не собирается.

— Сколько человек задействовано в поддержке? Есть ли у тебя преемник, если вдруг пришлось бы отказаться от поддержки reiser4?

Пока что я один. Все прежние разработчики ушли на заработки, а новых нет. Погрузиться в эту область непросто. Это целыми днями надо сидеть и корпеть за монитором. В цветущие годы обычно не до этого. Ну а когда человеку уже за тридцать, он хочет стабильной работы и денег. Где я ему их достану?

— Есть ли возможность лицензировать код reiser4 для применения в проприетарной ОС?

Я далёк от таких вопросов. Можно спросить у Ханса, если сильно интересно.

— Может ли reiser4 стать ФС по умолчанию в одном из следующих выпусков RHEL?

Это вопрос, скорее, к моему менеджеру: я не могу обсуждать планы Red Hat. Скажу только, что пока никому ничего не предлагал, а у меня никто ничего не спрашивал.

— Планируется ли портирование reiser4 на FreeBSD? Возможно, стоит подумать о

создании порта с помощью FUSE? Что ты думаешь о политике принятия изменений в ядро?

Вообще, портирование как таковое мне никогда не было интересно. Но я слышал, что FreeBSD — это операционная система, которая имеет академические корни (*University of California, Berkeley*). А это означает, что с большой долей вероятности мы найдём с разработчиками общий язык. Во всяком случае, там не будут при слове «алгоритм» смотреть на тебя с непониманием. В Линуксе же ключевым понятием является понятие патча. И существует комитет из определённых людей, которые решают (*основываясь вероятно на собственной интуиции, а также в большой степени на умении автора патча «ладить» с командой разработчиков ядра*), примут они этот патч в ядро, или нет. Мне такой подход не очень по душе: я закончил мехмат МГУ, а не МГИМО.

— С какими «подводными камнями» может столкнуться желающий попробовать reiser4 в повседневной работе? Как ты оцениваешь её стабильность?

Общий комментарий: за последние четыре года я не помню, чтобы кто-то терял данные на reiser4 разделе при исправно работающем железе. Ко мне обращалось несколько человек с жалобой на работу fsck. В конечном итоге все они получали и свои данные и работающий fsck.

Самое неприятное — это то, что может возникнуть необходимость отката на предыдущую версию ядра после апгрейда (*я не очень хорошо тестирую патчи для очередной версии*). Следующая неприятность — отсутствие утилиты дефрагментации. Также до сих пор живёт старый трудновоспроизводимый баг, приводящий к сообщениям о «key inconsistency». В любом случае, если вы решили связаться с reiser4, то непременно нужно запастись терпением. Если возникли проблемы, то надо послать багрепорт в лист рассылки, или же прямо мне (*если не владеете английским языком*). При этом не надо думать, что я мгновенно их решу: на reiser4 у меня есть время только после работы и выходные. Если я перестал отвечать на письма — не надо стесняться напомнить о себе еще раз. Ну а жаловаться на форумах — самый неэффективный способ решения проблем.

— Планируется ли создать утилиту для дефрагментации? Например, при использовании reiser4 на разделе с торрентами, получалось больше 11000 фрагментов на 700-мегабайтный файл, и никаким копированием не получалось сбить этот показатель хотя бы до нескольких сотен. При этом были ощутимы негативные последствия для производительности.

Да, планируется. Иметь такую утилиту важно. Менеджер транзакций Reiser4 использует смесь техник журналирования и copy-on-write. Последняя сама по себе уже означает фрагментацию. Для того, чтобы от неё избавиться, одиночного копирования может быть недостаточно: ведь свободное дисковое пространство тоже может быть фрагментировано. В общем случае утилита дефрагментации будет существенно улучшать ситуацию за несколько проходов дерева. С внешней фрагментацией можно бороться — это не есть приговор для ФС.

По поводу торрентов. Года три назад в Линуксе появился системный вызов fallocate(2), который призван предотвращать фрагментацию в подобных случаях. Приложение должно заранее указать смещение и размер куска в файле, а файловая система должна выделить под этот кусок (*как можно менее фрагментированное*) место на диске. Однако, reiser4 пока этот системный вызов не поддерживает. Сделать такую поддержку несложно, но в ближайшее время мне, скорее всего, будет не до этого.

— Есть ли проблемы со специфическим железом при использовании reiser4?

Не слышал о таких. Вроде как, всеядна.

— Будет ли реализована поддержка reiser4 для grub2 силами самих разработчиков reiser4?

Надеюсь, что будет. Это кропотливая работа, но она гарантировано увенчается успехом. Есть патч для grub-0.97. Основываясь на нём, чудесным образом можно организовать поддержку reiser4 для grub2. Недостаток имеющегося патча — это то, что загрузка не может идти через stage1_5 по той причине, что соответствующий бинарник получается слишком большой и не умещается в отводимые ему 62 сектора. А невозможность грузиться через stage1_5 означает, что каждый раз, когда на вашем разделе поработал дефрагментатор, нужно заново инсталлировать grub. В поддержке reiser4 для grub2 всё должно быть сделано хорошо. Модуль, грузящий btrfs с мультиустройств, уместился у меня в 62 сектора. Почему reiser4 туда не поместится?

— Возможен ли в будущем вынос плагинов в userspace? Есть ли вообще такие планы? Планируется ли создание инфраструктуры, которая могла бы загружать плагины как в пространстве ядра, так и в пространстве пользователя?

Вынос отдельных плагинов в userspace лишен особого смысла. Как они будут между собой взаимодействовать? Каждый плагин выполняет какой-либо сервис и, в свою очередь, просит другие плагины о каком-либо сервисе. Представьте, что плагину X, работающему в составе ядра понадобился какой-либо оперативный сервис, а предоставляющий его плагин Y работает в userspace. Ничего ведь хорошего при этом не получится? Динамическая загрузка плагинов как модулей ядра полезна, однако это не есть интересный и животрепещущий вопрос. Ну, давайте сделаем их динамически загружаемыми…

— Стоит ли задуматься о написании комплекса тестов, которые будут проверять ФС на прочность различными способами и показывать проблемы? Например, это мог бы быть набор скриптов на perl, который проводил бы агрессивные параллельные запись, считывание и удаление, показывал бы соответствие читаемых данных записанным, а также проводил бы проверку структуры ФС на предмет проблемных мест.

Это мечта многих иметь такие тесты. Чтобы через пол-часа их прогона можно было было бы с уверенностью делать очередной релиз. Скажу только, что здесь всё очень непросто. Написание всесторонних тестов на предмет выявления проблемных мест в программных продуктах — очень сложная задача. Да и тестирующий будет упираться в основном в регрессии других подсистем ядра. И либо исправлять их, либо ждать, когда кто-то другой их исправит.

— Как повлияли zfs/btrfs на reiser4?

Никак. На reiser4 частично повлияла разработка xfs (*техника «delayed allocation»*). В основном же использовались собственные наработки.

— Занимаешься ли ты непосредственно разработкой btrfs?

Отчасти по поручению работодателя. Я сделал поддержку btrfs в grub-0.97 (*с grub2 наши дистрибутивы не работают*). Не знаю, что поручат ещё. Вполне возможно, что ультрамодную фичу «data deduplication».

— Каково твоё мнение о текущем положении дел с btrfs по итогам недавней нашумевшей переписки с Мейсоном?

Почему ж «нашумевшей»? Обычная рабочая обстановка. Мне поручили исследовать btrfs на предмет её применимости в энтерпрайз-системах, ну я и нашёл сильную внутреннюю фрагментацию на тех моделях, где остальные ФС работают безупречно. Соответственно начал выяснять, ошибка это, или «фича». Правда, прошло уже пол-года а я до сих пор ничего вразумительного про алгоритмы btrfs так и не услышал. Какое тут может сложиться мнение? Понял только, что они хотят фичу «tail packing» файловой системы reiserfs, совершенно не понимая, как работают алгоритмы и структуры данных последней. Скажу только, что в B-деревьях понятие «tail packing» начисто лишено какого-либо смысла. И, более того, попытка размещать в таких деревьях айтемы переменного размера ведёт к неограниченной внутренней фрагментации. А Reiserfs **не использует** B-деревья и их общеизвестные модификации. Там применяются совсем другие алгоритмы (*изобретение российских ученых, между прочим*) — с них в начале 90-х началась история Namesys. И модифицировать их для балансировки «сверху вниз», как того требует дизайн btrfs, — задача не совсем тривиальная в отличии от классических B-деревьев.

Очень часто слышу о том, что маинтейнер btrfs Крис Мейсон, поработав в своё время в Namesys, как Дункан Маклауд позаимствовал оттуда весь позитивный опыт наработок. Я же пока что вижу только обратное. На ключах он почему-то сэкономил (*ключ в btrfs — 136 бит, в reiser4 — 192 бита*), но терабайты дискового пространства (*и оперативной памяти*) пользователей неудачной балнсировкой под откос пустил. Дополнительные поля ключа — это возможность по-разному группировать данные и метаданные. И что, это всё не нужно??? А балансировка сверху вниз — так это вообще, по-моему, сплошной компромисс: фазу squeeze балансировки, а также компрессию и шифрование данных отложить наподобие техники «delayed allocation» нельзя. А потом, мне кажется, что эти ребята упрутся в проблемы с масштабируемостью из-за невозможности организовать приличную схему блокировок на таком дереве. Скажу только, что гораздо выгоднее распределить «работу по дереву» между бóльшим количеством процессов, и пустить часть из них навстречу (*снизу вверх*), а не так, чтобы все они ломились в это дерево сверху через общий корень.

В общем, не знаю… Я, конечно, помогу, чем смогу, но тут такое дело: если проект основан на неудачных идеях, из него сложно сделать конфетку. К слову, вся история Namesys — это непрерывные контакты с академическими институтами (*МГУ, Институт программных систем РАН в Переславле-Залесском*). XFS — это тоже целая школа в Silicon Graphics. А Btrfs — это история чего? Пары низкоуровневых воркшопов? А как ещё назвать мероприятия, на которых анонсируются несуществующие фичи? В чудеса я давно перестал верить…

— Каким ты видишь будущее ФС? Какой у них будет функционал?

Файловая система — это подсистема, управляющая ресурсом «дисковое пространство». И все её «фичи» должны быть направлены на эффективное управление данным ресурсом. А это значит, что будущее файловых систем за более прогрессивными алгоритмами, т.е. за теми, которые лучше справляются с этой задачей. Однако, появляются новые физические носители, технологии чтения-записи, некоторые перемещаются из userspace в ядро (*атомарность, прозрачная компрессия, шифрование и пр.*). Существующие файловые системы становятся морально устаревшими: их дешевле переписать заново, чем приспособить к нововведениям. Файловые системы должны уметь «встречать» такие фичи. Не переписывать же их каждый раз заново… А для этого они должны обладать соответствующей технической базой.

Попытка создания такой базы была предпринята в reiser4: в отличии от своих предшественников она имеет полностью модульную структуру. В reiser4 все реализации методов file_operations, inode_operations, address_space_operations — это всего лишь тонкие прослойки — диспетчеры, которые решают, какому плагину (*модулю*) передать дальше управление. А каждый модуль реализует какой-либо абстрактный класс (интерфейс) определённой подсхемы интерфейсов, отражающей некоторую концепцию хранения (*мета*)данных.

Попытаюсь на пальцах объяснить, как всё это работает. Допустим, вы хотите реализовать функциональность btrfs (*снэпшоты и пр.*). Как известно, эта ФС использует транзакционнаую модель «copy-on-write», реализованную на базе балансировки дерева «сверху вниз». В этом её основное отличие от того, что в настоящее время предлагает Reiser4. Следовательно, мы должны создать новый плагин «cow» интерфейса «TMGR» (*transaction manager*), а также новый плагин «multi-root-tree» интерфейса «TREE» для дерева-хранилища, обладающего семейством корней (*«историей»*) и балансируемого сверху вниз. При этом последнее нужно снабдить своей схемой блокировок. Что касается TMGR, — это абстрактный класс для управления объектами, которые в следующей статье именуются «particles» (*понятие, дуальное примитиву «transcrash», статья находится здесь*).

Если присмотреться к менеджерам транзакций разных файловых систем (*на настоящий момент существует три типа таких менеджеров*), то несложно подметить некоторые их общие черты. А именно, в интерфейсе TMGR можно выделить набор из следующих основных методов:

- enter_context();

- try_capture();
- exit_context().

Первый и последний вызываются соответственно во время входа и выхода процесса из собственно файловой системы. Второй — во всех местах, где происходит модификация данных (*страниц или буферов*). Сейчас в reiser4 работает один-единственный TMGR-плагин, назовём его «jcow» (*симбиоз техник «journalling» и «copy-on-write» без сохранения истории*), метод ->try_capture() которого добавляет блок в т.н. «атом» (*специальное название для «particles» в reiser4*). А в нашем новоиспеченном плагине «cow» этот метод будет отпочковывать новый корень дерева-хранилища (*в коде btrfs соответствующая функция зовётся btrfs_cow_block*).

В качестве домашнего упражнения предлагаю понять, что именно в этом случае будет «атомами» (*т.е. должно отправляться на диск целиком*). За ликбезом можно обратиться к статье Охада Родеха «B-trees, shadowning, and clones».

Эти новые корни нужно уметь куда-то складывать и извлекать: если вы хотите фичу «writable snapshots», то они должны образовывать дважды индексированное множество. Но это не проблема. К примеру, в btrfs для этой цели используется отдельное «дерево корней».

Итого, нам нужно всего два новых плагина, чтобы получить функциональность btrfs. И действительно: а зачем нам что-то ещё? Плагины интерфеса FILE выбирают из дерева айтемы в соответствии с методами интерфейса TREE и не должны знать, как балансируется то или иное дерево. Плагины остальных интерфейсов (*NODE, ITEM, и пр.*) тоже остаются при деле: зачем нам менять формат узлов дерева для организации снэпшотов? Просто, наше «мультикорневое» дерево будет содержать разные внутренние узлы, ссылающиеся на одни и те же блоки.

Я не утверждаю, что программировать новые плагины интерфейсов TREE и TMGR — это работа для ленивых, но, поверьте, это намного проще, чем заново создавать файловую систему и сложнейшую утилиту fsck (*которая для reiser4 тоже, кстати, модульная*), ибо самый интересный момент здесь состоит в том, что имеющиеся плагины остальных интерфейсов не нужно учить работать с новыми членами семьи, а это значит, что отпадает необходимость в написании и отладке кода, процентное содержание которого будет стремиться к 100% (*при удачной организованной схеме интерфейсов вы успешно реализуете ещё не одну функциональность*).

Точно так же, при помощи плагинов, можно организовать и управление логическими томами как в ZFS или btrfs. Однако, тут я должен предостеречь: это уже будет так называемое смешение уровней (*layering violation*). Дело в том, что в Линукс управление томами осуществляется отдельной подсистемой (*lvm*), и попытка смешать её с файловой системой может окончиться плачевно: вас попросят эту функциональность убрать, и больше так не делать: здесь существует необъяснимая политика двойных стандартов: кому-то смешивать уровни дозволено (*к примеру, btrfs*), а в reiser4 это делать не приветствуется. Во всяком случае, вспомнив шквал обвинений в адрес reiser4 на тему layering violation, я не стал бы рисковать затраченными усилиями.

Подробности и остальные не менее интересные приложения модульной архитектуры можно будет найти в моей статье (*пока еще не вышла, будет анонсирована в листе рассылки reiserfs-devel mailing list*).

Итак, я бы обрисовал будущее локальных файловых систем в частности как «шлифовку» вот таких вот «внутренних» интерфейсов. На самом деле, если присмотреться, то можно заметить, что никакие они не внутренние. Это как в алгебре: если у вас какое-либо линейное пространство V распадается во (*внутреннюю*) прямую сумму подпространств, то можно пойти и по обратному пути: построить при помощи конструкции внешней прямой суммы пространство, которое будет изоморфно V. Ну, а раз они не внутренние, то это достояние уже всех файловых систем. Никаких проблем с VFS тут не возникает (*подробнее об этом в статье*). Вообще, здесь я усматриваю много аналогий между программными системами (*в большей степени это относится к системам хранения данных*) и такими понятиями гомологической алгебры, как модуль, градуировка, фильтрация, и др. которые мне кажутся очень полезными.

И последнее: про «фичи». Меня часто спрашивают о том, как написать плагин для reiser4. Причем, ответный вопрос, а что же он у нас будет реализовывать, зачастую ставит вопрошавшего в тупик. Мне не нравится сама идея поставить на поток производство «фич» для файловой системы с модульной архитектурой. Это дисциплина, а не массовая индустрия развлечений. Никто же не ставит на поток доказательства математических теорем…

Я считаю, что сначала должна быть полезная идея из области хранения информации (*к примеру, снэпшоты*). Не думаю, что таких идей может быть слишком много. С такими идеями — добро пожаловать. Мы подумаем, как выразить её на языке вложений, добавим, если нужно, новые интерфейсы в общую схему и напишем соответствующие плагины.

К сожалению, «фичи» зачастую высасываются из пальца: закон рынка (*что потребителя непременно нужно шокировать фичами*) не обошел стороной и этот «святой» сектор: за отсутствием каких-либо идей в области хранения данных файловые системы начинают «кипятить океан» и заниматься остальными проплаченными, но совершенно ненужными вещами.

pfactum, 23 ноября 2010 в 00:51