

СИСТЕМНЫЙ администратор

№9(70) сентябрь 2008
подписной индекс 20780
www.samag.ru

**Настраиваем терминальное решение
на базе Citrix Presentation Server 4.5
и тонких клиентов WYSE**



**На страже безопасности –
Software Restriction Policies**

**Новшества в Windows Server 2008:
стек TCP/IP**

**Используем check_relay в Sendmail
для борьбы со спамом**

**Работаем с данными
Active Directory из скриптов**

**Управление BerkeleyDB
и тонкая настройка Jabberd2**

**Точная геонавигация в городе
с помощью технологии Wi2Geo**

**Запускаем «САМО-ТурАгент»
под Linux**



Найди в журнале код.
Следуй инструкциям.
На кону настоящий
«АДМИНСКИЙ ПРИЗ»



12
лет

**НАША КОМПАНИЯ ПРЕДОСТАВЛЯЕТ
ЛИНИИ СВЯЗИ В САМЫХ НЕПРОХОДИМЫХ
МЕСТАХ МОСКВЫ**

**10 МБИТ - \$500, ВКЛЮЧЕНО МНОГО ТРАФИКА.
ANYTHING ELSE?**

ЗВОНИТЕ, ДОГОВОРИМСЯ!

г. Москва, Хлебный переулок 2/3, тел. 291-61-32, 202-61-43 (круглосуточно)
e-mail: support@redline.ru

5 ТЕНДЕНЦИИ

АДМИНИСТРИРОВАНИЕ

6 Устанавливаем и настраиваем терминальное решение на базе Citrix Presentation Server 4.5 и тонких клиентов WYSE

Первые этапы при вводе в строй терминального решения. Вопросы выбора продуктов, подготовки сетевой инфраструктуры, особенности установки терминальных продуктов Microsoft и Citrix.

Алексей Бережной

16 Новшества в Windows Server 2008: стек TCP/IP

В новой версии ОС Windows Server 2008 стек TCP/IP был полностью переработан. Рассмотрим, какие новшества появились в нем.

Андрей Бирюков

20 Используем check_relay в Sendmail для борьбы со спамом

Почтовые серверы, такие как Postfix и Exim, предлагают сравнительно прозрачные способы фильтрации соединений. Владельцы же Sendmail вынуждены попотеть, чтобы раскрыть весь потенциал этого МТА.

Сергей Супрунов

27 Jabberd2 – простой и нетребовательный к ресурсам XMPP-сервер. Часть 2

Управление базами данных BerkeleyDB и тонкая настройка Jabberd2.

Михаил Кондрин

34 Работаем с данными Active Directory из скриптов
Часть 1. Применение Bourne Shell

В корпоративной среде при разработке скриптов рано или поздно возникает вопрос: «А неплохо было бы иметь базу всех сотрудников с именами, должностями, адресами и номерами ICQ...». Такая база уже есть. Ее за нас придумала Microsoft. Это Active Directory. Нужно только уметь с ней общаться.

Рашид Ачилов

40 Панель управления хостингом ISPConfig

Обзор решения.

Сергей Яремчук

44 Лебедь, рак да щука: САМО-Софт, wine, Etersoft

Запускаем под Linux «САМО-ТурАгент» – продукт для автоматизации турфирмы.

Сергей Барановский

50 Обзор ALT Linux 4.0 Children

Знакомимся с новинкой – бета-версией необычного дистрибутива, предназначенного для детского творчества – ALT Linux 4.0 Children.

Сергей Яремчук

53 ЗАКОН ЕСТЬ ЗАКОН

БЕЗОПАСНОСТЬ

54 На страже безопасности – Software Restriction Policies

Эффективно используем технологию SRP.

Вадим Поданс

ИНТЕРВЬЮ

60 Точная геонавигация в городе с помощью технологии Wi2Geo

Интервью с Денисом Алаевым, автором идеи.

Антон Борисов

ЧЕЛОВЕК НОМЕРА

66 Ричард Столлман: свобода без ограничений

Пожалуй, нет в мире человека, знакомого со свободным программным обеспечением и не знающего Ричарда Столлмана – человека, подарившего миру идею бесплатного ПО с открытым исходным кодом, яркого противника авторского права.

Дмитрий Мороз

ПРОГРАММИРОВАНИЕ

70 Распределенная система контроля версий Bazaar

Основные концепции Bazaar.

Дмитрий Васильев

77 Готовимся к переходу на PHP 5.3

Для плавного перехода с PHP 5.2 на PHP 6 было решено разработать промежуточную версию PHP 5.3, которая будет содержать большинство улучшений, разработанных для PHP 6.

Александр Майоров

АДМИНИСТРИРОВАНИЕ «1С»

86 Открываем базу данных одним щелчком мыши

Файловое расширение *.MD по умолчанию никак не обрабатывается операционной системой, но это досадное недоразумение можно исправить, добавив себе дополнительный функционал для комфортной работы.

Андрей Луконькин

87 «1С:Сценарное тестирование»

В сентябре вышла ознакомительная версия нового программного продукта, позволяющего повысить работоспособность и отказоустойчивость конфигураций «1С:Предприятие 8».

Андрей Луконькин

РЕТРОСПЕКТИВА

88 Сквозь тернии к звездам: история компании Hewlett-Packard

Их было двое. Одного звали Билл, другого – Дэйв. Они окончили университет в начале тридцатых годов прошлого века, и были инженерами. Эти парни хотели заниматься технологиями. Теперь фамилии Hewlett и Packard знает весь мир.

Илья Александров

92 КНИЖНАЯ ПОЛКА

94 СИСАДМИН ТОЖЕ ЧЕЛОВЕК

19, 65, 91 BUGTRAQ

Код для участия в розыгрыше призов. Следуйте инструкциям на стр. 2.



Скорость. Надежность. Поддержка.



Три шанса выиграть

Дорогие участники розыгрыша призов!

Для Вас есть две новости: одна – хорошая и вторая – тоже хорошая.

1. В сентябрьском номере находится последний код, необходимый для участия в розыгрыше «Админский Приз РАЗ».
2. Всем зарегистрированным участникам розыгрыша автоматически засчитан код из июльского журнала.

Итоги розыгрыша «Админский Приз РАЗ» будут опубликованы на сайте журнала www.samag.ru 27 ноября 2008 года.

Внимание! Активировать коды для участия в первом этапе розыгрыша необходимо до 20 ноября 2008 года.

Не останавливайтесь на полпути! Ваш следующий шанс выиграть приз – стать участником розыгрыша «Админский приз ДВА». Для участия Вам понадобятся коды из журналов №10, 11, 12 за 2008 год.

Админский Приз

Админский Приз «РАЗ»

- ☒ Коммуникатор
- ☒ Учебные курсы
- ☒ Программные продукты
- ☒ Виртуальные выделенные серверы

Админский Приз «ДВА»

- ☒ Коммуникатор
- ☒ Учебные курсы
- ☒ Программные продукты
- ☒ Виртуальные выделенные серверы

Админский Приз «ТРИ»

- ☒ Ноутбук
- ☒ Учебные курсы
- ☒ Программные продукты

Все на розыгрыш призов...

**СИСТЕМНЫЙ
администратор**

Более подробную информацию о сроках и правилах проведения розыгрышей призов смотрите на сайте журнала – www.samag.ru

LINBOX RESCUE SERVER



LINBOX RESCUE SERVER: полный контроль за ИТ-инфраструктурой

Linbox Rescue Server (LRS) это пакет программ, предоставляющий все функции надлежащего локального или удаленного управления ИТ системой. Он может быть использован как в одном подразделении, так и в десятках, размещая, контролируя и исправляя все (в основном Windows и GNU/Linux) компьютеры в сетях LAN и WAN. LRS может установлен на машину с GNU/Linux с любой поддерживаемой локальной сетью. Администратор может управлять через Веб-браузер с любого локального или удаленного компьютера. Он использует систему загрузки PXE и не требует установки специализированного клиентского ПО на машины.

- Легкость администрирования за счет использования единой мультифункциональной Веб-консоли. Использование зашифрованного туннеля и системы авторизации гарантирует контроль и конфиденциальность передачи через локальную или глобальную сеть.

- Резервирование и восстановление систем с Windows или GNU/Linux за несколько минут (примерно 10-15 минут для Windows+Office XP). Резервирование системы ведется в виде полного образа на жестком диске, который содержит операционную систему, программное обеспечение и параметры. Единственный образ (master) может быть использован для множества ПК, благодаря пост-инсталляционным особенностям (например введение компьютеров с Windows в домен). Вы можете выбрать какие разделы вы хотите резервировать или восстановить. Копия может быть сохранена на жестком диске, CD или DVD.

- Резервирование и восстановление файлов пользователей как под Windows и GNU/Linux, так и под Unix. Инкрементные или полные резервные копии автоматически сохраняются на жестких дисках и/или лентах для любых созданных или модифицированных файлов с заданной частотой. Как правило вы можете сохранить непрерывные копии за 30 дней на дисках сервера LRS. Система сжатия и различные функциональные особенности снижают объем требуемого места для сохранения данных до 1/3 от исходного. Восстановление очень легко и требует всего лишь нескольких кликов в Веб-интерфейсе.

- Полный учет программного обеспечения и оборудования. Сервер осуществляет полную опись машин и их окружения для всех подключенных компьютеров 24 часа в сутки: Опись оборудования (процессор, карт, памяти жесткого диска, числа дисков...), периферии (принтеры, сканеры и т.п.) а также программного обеспечения (тип, версия, редакция и т.п.) и сетевой информации (IP адреса, выходы и т.п.). Администратор может получить предупреждение по электронной почте при изменениях между двумя сессиями учета.

- Удаленный контроль за клиентскими ПК. Администратор может воспользоваться безопасным доступом к клиентской машине (через ssh туннель в случае необходимости), после его приглашения пользователем.

Компания Мандрива.Ру

телефон/факс: +7-499-271-49-55

e-mail: info@mandriva.ru

web: www.mandriva.ru

http://www.mandriva.ru/resheniya/produkty/linbox_rescue_server

Google выпустила свой веб-браузер – Google Chrome

Пожалуй, самым заметным событием начала осени стало заявление компании Google о намерении выпустить собственный веб-браузер с открытым кодом – Chrome (<http://www.google.com/chrome>).

1 сентября в официальном блоге Google появилось сообщение о том, что 2 сентября, компания одним простым и очевидным шагом буквально ворвется на нестабильный в последние годы рынок веб-браузеров. Обещание в Google сдержали, и миллионы пользователей бросились изучать и тестировать новый продукт интернет-гиганта. И вскоре исследователи из Net Applications заявили, что Chrome удалось за несколько часов после официального запуска завоевать 1% рынка браузеров, достигнув пика в 9,7% на сайте Computerworld на следующий день после дебюта (3 сентября).

В Google объяснили причину выпуска своего браузера следующим образом: «Мы уверены, что можем принести пользу в этой области и в то же время способствовать инновациям в Web». Браузер Google Chrome уже долгое время использовался внутри компании, и теперь его выпустили для широкой публики.

Среди технических подробностей о Google Chrome можно выделить заимствование компонентов из браузера Mozilla Firefox и движка рендеринга веб-страниц Apple WebKit, а также использование нового открытого JavaScript-движка V8 (его исходный код был вскоре опубликован на Google Code). Одной из ключевых особенностей стала изоляция вкладок браузера друг от друга («падение» одного из табов никак не отразится на работе других).

Первый бета-релиз Google Chrome пока доступен только для операционной системы Microsoft Windows (в Google объяснили это желанием получить максимальный отклик как можно быстрее), но в дальнейшем появятся сборки и для других платформ (Mac OS X, Linux). Пока в сети можно найти только рекомендации по самостоятельной компиляции Chrome из исходников для этих систем.

Sun наращивает активность в области Open Source

Начало сентября ознаменовалось спорным известием о том, что Михаэль Видениус (Michael «Monty» Widenius), один из основателей MySQL AB и главный автор оригинальной версии популярнейшей СУБД с открытым кодом MySQL, покидает Sun Microsystems. Однако сие событие ничуть не помешало известной IT-компании одновременно с этим выстрелить целой чередой Open Source-анонсов.

4 сентября Sun объявила о выпуске новой версии своего программного решения в области виртуализации для десктопов – Sun xVM VirtualBox 2.0.0. В пресс-релизе отмечается, что «программное обеспечение xVM VirtualBox – это первый крупный гипервизор с открытым кодом, поддерживающий наиболее популярные операционные системы (в роли хостов), среди которых Mac OS X, Linux, Windows, Solaris/OpenSolaris». Среди изменений, представленных в VirtualBox 2.0, поддержка 64-разрядных гостевых операционных систем (для 64-битных хостов), перевод GUI с Qt3 на Qt4, новый родной интерфейс для Mac OS X «Leopard»,

поддержку асинхронного ввода/вывода для SATA (NCQ), поддержку образов VHD (Microsoft Virtual PC).

Уже 10 сентября Sun Microsystems развила тему виртуализации, запустив новый Open Source-проект, нацеленный на улучшение возможностей решения Sun xVM Server. Анонсированный xVMserver.org стал инициативой Sun по созданию Open Source-сообщества на базе ее виртуализационного программного обеспечения xVM Server. На появившемся веб-сайте разработчики получили возможность скачать первый архив исходного кода Sun xVM Server и внести свой вклад в процесс совершенствования продукта. Объявление о запуске xVMserver.org состоялось вместе с сообщением о доступности программного обеспечения Sun xVM Server и Sun xVM Ops Center 2.0, которые в Sun называют «ключевыми компонентами в ее всеобъемлющей стратегии по виртуализации».

Кроме того, 9 сентября Тим Брэй (Tim Bray), директор по веб-технологиям в Sun, представил в своем блоге начало бета-тестирования Project Kenai – сервиса Sun для хостинга Open Source-проектов, позиционируемый компанией как «Больше, чем просто кузница», апеллируя к SourceForge (от англ. forge – кузница). Как пояснил Ник Сейгер (Nick Seiger), разработчик сервиса, одной из причин, побудивших к созданию Kenai, стала потребность в демонстрации возможностей создания проектов, работающих поверх традиционных стеков LAMP/SAMP. В этом заявлении автор ссылается на разработки Glassfish, JRuby и Ruby on Rails.

Подготовил Дмитрий Шурупов
по материалам www.nixp.ru

Zlock

КОНТРОЛЬ ДОСТУПА к USB УСТРОЙСТВАМ



ЗАЩИТА ОТ ВНУТРЕННИХ УГРОЗ

www.securit.ru

Устанавливаем и настраиваем терминальное решение на базе Citrix Presentation Server 4.5 и тонких клиентов WYSE

Алексей Бережной

В статье рассказывается о первых этапах при вводе в строй терминального решения на базе Citrix Presentation Server 4.5. Затрагиваются вопросы выбора продуктов, подготовки сетевой инфраструктуры, а также особенности установки терминальных продуктов Microsoft и Citrix.

Так уж получилось, что наш отдел оказался в численном меньшинстве. Два системных администратора (по UNIX и Windows-системам соответственно) и один инженер технической поддержки на парк компьютеров более 200 машин — это совсем немного. Помимо всего стартовал новый проект по переходу на новую СУБД. Поэтому нужно было искать способ в кратчайший срок максимально сократить издержки на обслуживание сетевой ин-

фраструктуры. Мы остановились на терминальном решении: тонкие клиенты и терминальный сервер.

Преимущества использования тонких клиентов

- длительный срок работы клиентских устройств;
- большой срок морального устаревания оборудования;
- быстрота развертывания;
- простота обслуживания;
- повышение уровня безопасности;
- упрощение процесса резервного копирования;
- простота учета компьютерной техники;
- невысокая стоимость обслуживания тонких клиентов;
- низкое энергопотребление;
- отсутствие шума от вентиляторов в системных блоках;
- отсутствие пыли от компьютеров;

- экономия пространства на столе и под столом, а также возможность использования любой офисной мебели;
- стоимость тонкого клиента.

Длительный срок работы клиентских устройств

Действительно, срок наработки на отказ тонкого клиента несравненно выше, чем персонального компьютера, снабженного дисковой подсистемой, вентилятором на процессоре и блоке питания, съемными элементами, например, сетевой или видеокартой. В тонком клиенте отсутствуют масса деталей, вызывающих какие-либо сбои. Нет операционной системы, время от времени «радующей» пользователя «синим экраном смерти». Нет вращающихся деталей, как то: вентилятор на процессоре или видеокарте. Нет жесткого диска, начинающего сбоить в самый неподходящий момент. Нет дополнительных плат расширения, качающихся в слотах расширения. Все, что нужно, содержится в маленькой коробочке, которую, скорее всего, никогда не придется вскрывать.

Большой срок морального устаревания оборудования

С тонким клиентом не нужно заботиться об устаревании оборудования, о том, что клиентский системный блок больше не соответствует техническим требованиям новой операционной системы или прикладному программному обеспечению, потому сам по себе тонкий клиент содержит минимум функций, как правило, достаточных только для подключения к терминальному серверу.

Быстрота развертывания

Сколько времени уходит на подготовку обычной офисной рабочей станции? Если устанавливать с «нуля», начиная с операционной системы и драйверов оборудования и заканчивая офисными приложениями, то примерно около двух часов. Даже при применении средств быстрого развертывания, например, из образа, созданного посредством Acronis True Image или Norton Ghost, все равно процедура займет примерно около часа. В случае с тонким клиентом достаточно достать оборудование из коробки и подключить кабели к разъемам. Что же касается прикладного программного обеспечения, то оно устанавливается один раз на терминальный сервер.

Простота обслуживания

Достаточно выполнить минимальные настройки: задать первоначальные параметры подключения. В случае неисправности тонкие клиенты легко отвезти в гарантийную мастерскую.

Повышение уровня безопасности

Действительно, у тонкого клиента нет жесткого диска с операционной системой, который можно снять и переписать с него данные или взломать операционную систему и получить доступ к компьютеру. Крайне затруднен процесс кражи информации при помощи USB-носителей и других средств переноса данных.

Резервное копирование

В случае с тонкими клиентами не стоит беспокоиться по поводу данных, сохраненных на жестких дисках пользова-

телей. Все, что нужно для работы, теперь хранится на терминальном, или файловом сервере. Соответственно процесс резервного копирования упростился в десятки раз.

Простота учета компьютерной техники

Не нужно использовать дополнительное ПО для сбора информации об аппаратном и программном обеспечении рабочих станций. Нет нужды вскрывать системные блоки, чтобы ознакомиться с оборудованием, которое не удалось протестировать программно. Достаточно переписать серийные номера тонких клиентов.

Невысокая стоимость обслуживания тонких клиентов

Во-первых, как уже говорилось, практически нет деталей, которые выходят из строя. Во-вторых, нет сложного ПО,

Лицензионное ПО — это высокое качество, надежность и репутация!

- ❖ ВЫСОКОКВАЛИФИЦИРОВАННЫЙ КОНСАЛТИНГ
- ❖ ПОДБОР ПО ДЛЯ БИЗНЕСА
- ❖ АНАЛИЗ И АУДИТ
- ❖ КОРПОРАТИВНОЕ ЛИЦЕНЗИРОВАНИЕ
- ❖ ТЕХНИЧЕСКАЯ ПОДДЕРЖКА

Softway.ru — авторизованный партнер крупнейших производителей программного обеспечения:

- ❖ Microsoft Gold Certified Partner,
- ❖ Oracle, Autodesk, IBM, Adobe, Corel, 1C, Symantec, Kerio, Лаборатория Касперского, Eset, ABBYY, Acronis и других.

Москва, ул. Б. Грузинская, д. 36А, стр. 5
(495) 987 10 50

Новосибирск, ул. Восход 20, офис 305
(383) 254 03 05



Softway.ru

лицензионное
программное
обеспечение

www.softway.ru

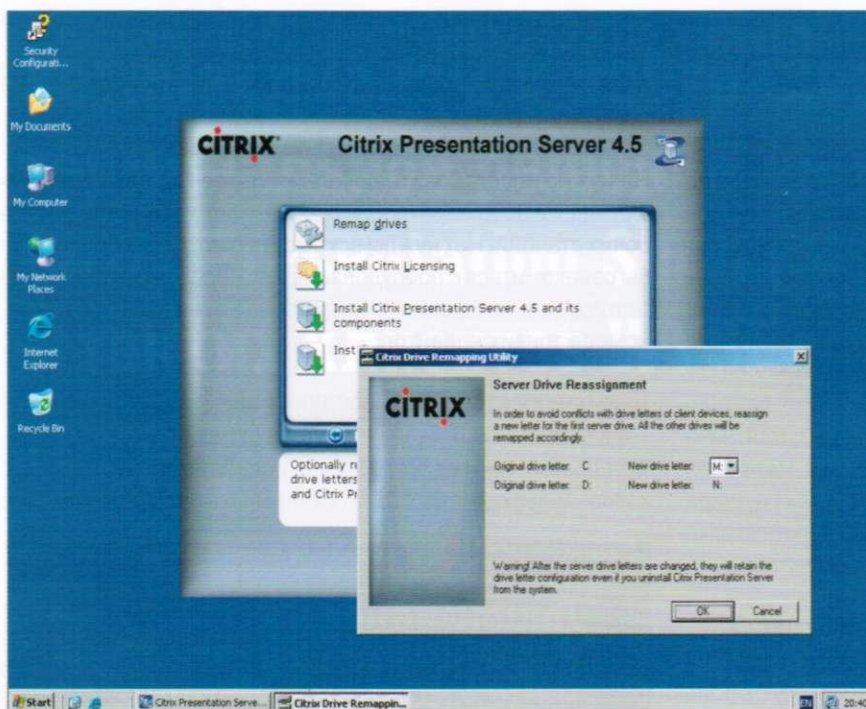


Рисунок 1. Окно установки Citrix «Переназначение дисков» (Remapping)

операционной системы и прочих вещей, на которые в большинстве своем тратятся основные усилия сотрудников технической поддержки. Добавьте к этому крайне низкую вероятность вмешательства пользователя в настройки терминала, и получится весьма впечатляющая картина «непотопляемой» ИТ-структуры. Это выгодно не только для крупных компаний, но и для небольших фирм, которые впоследствии смогут окончательно перейти на аутсорсинг.

Низкое энергопотребление

К сожалению, в России в большинстве компаний пока не обращают внимания на реальные убытки от перерасхода электроэнергии. Но достаточно представить себе крупный офис с численностью от 100 компьютеров и выше, чтобы понять, что даже небольшая экономия в 100 Ватт на каждое устройство может вылиться в реальные десятки, а то и сотни киловатт/час, когда речь идет обо всем офисе в целом.

Отсутствие шума от вентиляторов в системных блоках

Само по себе большое достижение. Бывают забавные случаи, когда, например, сотрудники одной компании, где был произведен переход на терминальные решения, пожаловались руководству на необычную тишину

в кабинетах. Создавалось впечатление, что как будто не хватало чего-то привычного. Выход нашлся быстро — по внутренней громкоговорящей связи стали транслировать тихую музыку.

Отсутствие пыли от компьютеров

Обычный офисный компьютер не просто работает, как плохой пылесос. Крупные частички пыли всасываются через щели системного блока, как мельницами, измельчаются вентиляторами и вылетают наружу, оседая на поверхностях. Дышать этой пылью — далеко не самая лучшая оздоровительная процедура.

Экономия пространства на столе и под столом, а также возможность использования любой офисной мебели

Не секрет, что некоторые образцы современной офисной мебели не в состоянии выдерживать вес современных системных блоков. Например, некоторые столы из стекла и/или пластика просто разваливаются под тяжестью обычного персонального компьютера. Возможность использовать маленький легкий тонкий клиент позволяет покупать любую офисную мебель.

Стоимость тонкого клиента

Как правило, стоимость тонкого клиента примерно на 100-150\$ ниже сто-

имости обычного персонального компьютера (не считая лицензии на программное обеспечение). На первый взгляд разница в цене не очень большая, но если речь идет об организации, чей парк насчитывает более 100 персональных компьютеров, выгода может оказаться весьма внушительной.

Минусы использования тонких клиентов

- чрезмерная централизация сетевой инфраструктуры;
- утрата некоторых пользовательских функций;
- утрата статуса «большого босса».

Чрезмерная централизация сетевой инфраструктуры

Когда роль клиентов сведена к минимуму функций, роль терминального и файлового серверов значительно возрастает. Справедливости ради стоит отметить, что в случае с обычными ПК все равно существует большая зависимость от серверной части. Например, все от того же файлового сервера, от контроллера домена, от сервера DNS и т. д.

Утрата некоторых пользовательских функций

Тонкие клиенты в классическом понимании не содержат в своем составе таких устройств как DVD-ROM и т. п. вещей. Хотя в то же время процесс просмотра видео, прослушивания аудиоинформации вполне осуществим на большинстве моделей современных тонких клиентов.

Утрата статуса «большого босса»

Не секрет, что ПК может способствовать удовлетворению тщеславия и личных амбиций. Зачастую чем выше ранг сотрудника в компании, тем более дорогой системный блок стоит у него под столом. Хотя с точки зрения финансовой целесообразности такая практика измерения статуса крайне негативно отражается на финансовом положении большинства компаний, а также на информационной безопасности и отказоустойчивости. Гораздо выгоднее потратить деньги, сэкономленные на амбициях менеджеров среднего звена, на модернизацию серверного оборудования или, к при-

меру, на создание мощной системы резервного копирования.

Выбор тонкого клиента

Тонкие клиенты выпускает множество компаний. Среди них HP, Sun Microsystems, WYSE, а также огромное количество разных мелких сборщиков. В силу вышеописанных причин нам была нужна самая простая и надежная модель.

Кроме того, по нашему замыслу он должен поддерживать протокол ICA для подключения к Citrix Presentation Server и обладать минимумом пользовательских функций, по принципу — меньше функций, меньше возможностей сбить настройки. Это позволит сократить время на обслуживание заявок от пользователей, переведя все программное обеспечение на терминальный сервер.

Мы решили остановиться на модели от WYSE — S10 (<http://wyse.ru/products/winterm/S10/index.php>). Простота исполнения, эргономичность, минимум функций и невысокая цена — эти показатели определили наш выбор.

Честно признаться, мы немного колебались в сторону более продвинутых моделей, таких как WYSE S50 и V50. В этих моделях нас привлекала возможность использования протокола X Windows, а значит, возможность использования Linux в качестве операционной системы для терминальных решений. Но, имея ряд нерешенных вопросов, в частности, совместимости используемого ПО, мы решили отказаться от этой идеи и сосредоточиться на терминальных решениях от Microsoft и Citrix.

Выбор операционной системы терминального сервера

Как уже говорилось, нам очень хотелось остановить свой выбор на сервере под управлением бесплатного дистрибутива Linux и системой X Windows. Главным преимуществом данного решения было отсутствие необходимости покупать операционную систему. Однако данный выбор, во-первых, потребовал бы приобретения более сложных дорогих тонких клиентов, во-вторых, могли возникнуть проблемы с совместимостью Linux с ря-

дом прикладных программ, таких, как банк-клиенты, рукописные СУБД и т. п. Так как сроки на решение задачи были весьма ограничены, пришлось остановить свой выбор на терминальном решении, основанном на продуктах Windows и Citrix.

Еще одно дополнительное условие, которое определило наш выбор, — пользователи не должны были «замечать подмены» при работе с виртуальными рабочими столами. То есть максимально сократить и упростить процесс адаптации рядовых сотрудников в новой IT-структуре. Поэтому в итоге, взвесив все «за» и «против», мы остановились на операционной системе Windows Server 2003 R2 Standard Edition x64.

Выбор аппаратного обеспечения терминального сервера

Первоначально предполагалось подключить от 30 до 100 пользователей. Поэтому решено было использовать один сервер в ферме с последующим подключением новых серверов по мере возрастания нагрузки. Конфигурация сервера:

- серверная платформа Intel SR2500ALLXR;
- материнская плата S5000PAL;
- 2 процессора Xeon(R) CPU E5440 @ 2.83 ГГц 16 Гб RAM;

- дисковый массив RAID10 — 4x250 Гб;
- контроллер SCSI/RAID — Intel Embedded Server RAID Technology II;

Решение проблем с печатью

Если раньше пользователи распечатывали на локальных принтерах, подключенных к персональным компьютерам, то при переходе на тонкие клиенты возникла необходимость решения вопросов сетевой печати. Поэтому появилась необходимость значительную часть нагрузки перенести на имеющиеся сетевые МФУ (многофункциональные устройства). Для оставшихся локальных принтеров были куплены принт-серверы для дальнейшего подключения их к терминальному серверу в качестве сетевых печатающих устройств.

Почему Citrix?

На начальном этапе можно было остановиться на Microsoft Terminal Server. Закупленные тонкие клиенты WYSE S10 умеют работать по RDP-протоколу, что может быть вполне достаточно. Citrix Presentation Server был выбран за возможность масштабирования (серверы объединены в единую ферму, производительность которой можно увеличивать за счет подклю-

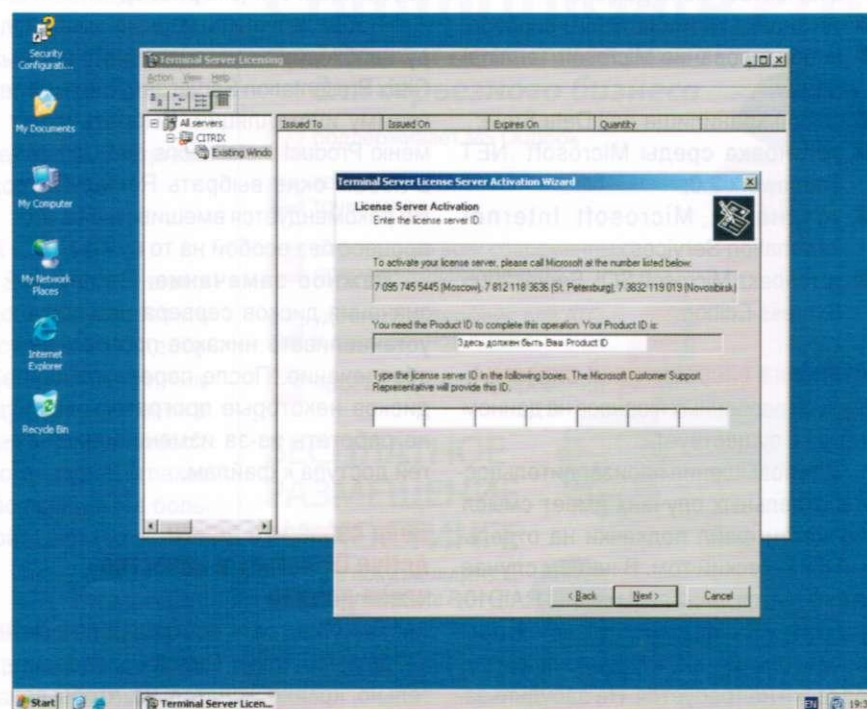


Рисунок 2. Окно активации Terminal Server Licensing Server Activation Wizard

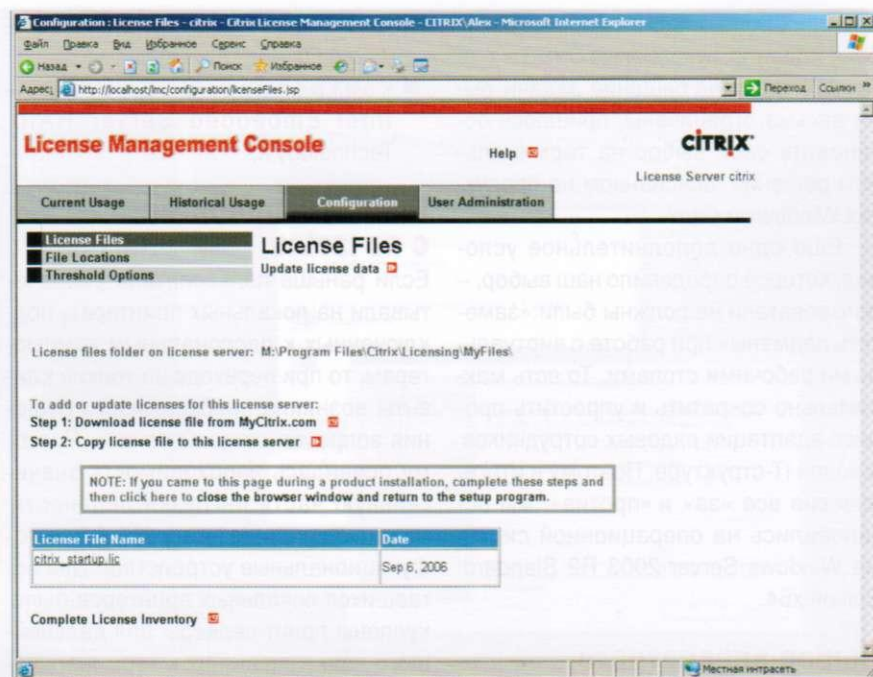


Рисунок 3. Консоль Citrix License Management

чения новых серверов). Также выглядели весьма привлекательно технологии публикации программного обеспечения посредством технологии Data Streaming.

Подготовка к установке аппаратного обеспечения Citrix

- установка операционной системы;
- переназначение дисков;
- ввод сервера в домен Active Directory в качестве члена домена;
- установка терминального сервера;
- лицензирование Microsoft Terminal Server;
- выбор хранилища для Data Store;
- установка среды Microsoft .NET Framework 2.0;
- установка Microsoft Internet Information Services;
- установка Microsoft SQL Server 2005 Express Edition.

Установка операционной системы

Никаких особенных нюансов на данном этапе не существует.

Для повышения производительности в отдельных случаях имеет смысл перенести файл подкачки на отдельный физический том. В нашем случае используется дисковый массив RAID10, который обладает достаточной производительностью, и переноса файла подкачки не требуется. Не забудьте зарегистрировать операционную систе-

му и установить все необходимые обновления.

Переназначение (Remapping) дисков

Для того чтобы пользователь при открытии терминальной сессии смог подключить свои локальные дисковые тома, используя не привычные буквы C, D, E и т. д. логическим томам сервера присваиваются другие буквы. По умолчанию в этом случае используются буквы, начиная с M (см. рис. 1).

Чтобы запустить данную процедуру, необходимо с установочного диска Citrix Presentation Server запустить программу инсталляции. Выбрать пункт меню Product installations and Updates, в новом окне выбрать Remap Drive. Не рекомендуется вмешиваться в этот процесс без особой на то нужды.

Важное замечание. До переназначения дисков сервера не следует устанавливать никакое программное обеспечение. После переназначения дисков некоторые программы могут не работать из-за изменившихся путей доступа к файлам.

Ввод сервера в домен Active Directory в качестве члена домена

Так как у нас сеть построена на основе домена Active Directory, следовательно, крайне желательно, чтобы все компьютеры являлись членами домена.

Справедливости ради стоит отметить, что Citrix Presentation Server предполагает установку и в качестве одиночного сервера, не входящего в домен, но такие случаи на практике применяются в последнее время все реже и реже.

Важное замечание. Крайне нежелательно использовать терминальный сервер в качестве контроллера домена. И хотя порой соблазн очень велик (особенно в плане экономии на лицензиях программного обеспечения), совмещать терминальный сервер и контроллер домена на одном сервере — это очень плохая идея. Мало того, что снижается производительность, еще и все терминальные пользователи получают прямой доступ на контроллер домена, что противоречит любым здравым требованиям информационной безопасности и обеспечения отказоустойчивости.

Установка терминального сервера

Citrix Presentation Server требует установленного Microsoft Terminal Server с требуемым количеством лицензий.

Важное замечание. Перед установкой Terminal Server необходимо выключить удаленный доступ к рабочему столу. Для этого правой кнопкой мыши необходимо щелкнуть по значку «Мой Компьютер» (My Computer), далее из появившегося меню выбрать «Свойства» (Properties), далее перейти на вкладку «Удаленное использование» (Remote) и снять отметку на опции «Включить удаленный доступ к рабочему столу» (Allow users remotely to this computer).

Устанавливать Terminal Server лучше всего из окна «Управление Сервером» (Manage Your Server). Вызвать его можно из меню «Панель управления (Control Panel) → Администрирование (Administrative Tools) → Управление данным сервером (Manage Your Server)». После установки сервер автоматически перезагрузится. По завершении перезагрузки можно проверить с рабочей станции удаленный доступ к рабочему столу.

Лицензирование Microsoft Terminal Server

Этот вопрос заслуживает отдельного рассмотрения. В первую очередь необходимо установить сервер лицензий (Terminal Server Licensing). Сделать это

можно из меню «Панель Управления (Control Panel) → Установка и удаление программ (Add and Remove Programs) → Добавить /удалить компоненты Windows (Add/Remove Windows Components) → Лицензирование сервера терминалов (Terminal Server Licensing).

Далее необходимо запустить MMC-консоль Terminal Server Licensing и, выбрав нужный сервер лицензий (в нашем случае он будет один), кликнуть правой кнопкой мыши. В появившемся меню выбрать «Активировать Сервер» (Activate Server). После чего в появившемся окне приветствия Terminal Server Licensing Server Activation Wizard нажать кнопку Next (далее). В следующем окне предлагается выбрать способ активации. Я выбирал по телефону (Telephone). В следующем окне появилась информация о необходимых телефонах, а также о Product ID, необходимом для активации (см. рис. 2). После звонка по указанному телефону меня соединили с милой девушкой, которая приняла от меня мой Product ID и продиктовала в ответ License Server ID, который я внес в пустые ячейки. Далее аналогичным образом активируются лицензии на подключение к терминальному серверу. Способ активации по телефону тем и хорош, что если что-то не знаете, вам обязательно помогут.

Выбор хранилища для Data Store

Citrix Presentation Server нуждается в средствах СУБД для хранения служебной информации. Для этих целей могут быть использованы:

- база данных MS Access;
- Microsoft SQL Server 2005 Express Edition;
- Microsoft SQL Server, Oracle database, and IBM DB2.

Третий пункт отпал сразу, так как тратить деньги на покупку еще одной лицензии мощной СУБД, такой как MS SQL Server или Oracle, не представлялось рентабельным.

Первый пункт отпал по той же причине (необходимость лицензии). К тому же связываться с «плоскими» (не SQL) базами данных типа Access не хотелось из-за малой производительности, трудностей с резервным копированием и т. д.

Остался вариант с Microsoft SQL Server 2005 Express Edition, что и определило дальнейший ход работы.

Установка Microsoft .NET Framework 2.0

Для нормальной работы Microsoft SQL Server 2005, включая версию Express Edition, вам понадобится установить Microsoft .NET Framework 2.0. Для счастливых обладателей дистрибутива Microsoft Windows Server 2003 версии R2 есть хорошая новость – данный продукт уже включен в дистрибутив. Обладателям более ранних версий Microsoft Windows Server 2003 придется скачать установочный пакет Microsoft .NET Framework 2.0 с сайта Microsoft (что в большинстве случаев не является проблемой – размер пакета составляет 22,4 Мб).

Установка Microsoft Internet Information Services

Для работы Microsoft SQL Server 2005 Express Edition и Citrix Presentation Server необходимо, чтобы на сервере был установлен Microsoft IIS. С установкой данного продукта также

не должно возникнуть особых трудностей, так как данный пакет уже включен в дистрибутив Microsoft Windows Server 2003. Аналогично другим устанавливаемым продуктам заходим в «Панель Управления» (Control Panel) и посредством апплета «Установка и удаление программ» (Add and Remove Programs) устанавливаем необходимые службы.

Установка Microsoft SQL Server 2005 Express Edition

Важное замечание. Microsoft SQL Server 2005 Express Edition поставляется в двух редакциях Microsoft SQL Server 2005 Express Edition и Microsoft SQL Server 2005 Express Edition with Advanced Services. В первом случае вы получите только непосредственно систему управления базами данных без каких-либо дополнительных пользовательских функций. Во втором случае помимо непосредственно «движка» (Engine) Microsoft SQL Server 2005 Express Edition вы получаете дополнительные возможности для построения отчетов, полнотекстового поиска, ну и конечно, консоль управления SQL Server 2005 Management Studio Express. Несмотря на значительную разницу в объеме скачиваемого файла (53 Мб против 234 Мб), мы остановили свой выбор на втором варианте, чтобы впоследствии иметь дополнительный инструмент для просмотра, резервного копирования и других манипуляций с базами данных. Хотя для экономии трафика, места на диске, системных ресурсов и т. д. можно использовать и первый вариант. Перед установкой необходимо уяснить некоторые детали.

Как известно, MS SQL начиная с MS SQL 2000 может быть установлен как экземпляр по умолчанию (Default)

Ищете надежное решение для корпоративной почты?

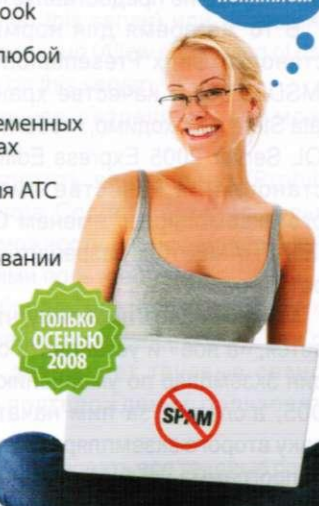
CommuniGate Pro authorized reseller

простое и понятное решение для серьезного бизнеса

- поддерживает MS Outlook
- веб-доступ к почте из любой точки мира
- работает во всех современных операционных системах
- удобный интерфейс для АТС и электронной почты
- прост в администрировании

БЕСПЛАТНОЕ РАЗМЕЩЕНИЕ НА ХОСТИНГЕ

при покупке лицензии CommuniGate Pro от 75 пользователей



подробности www.rusonyx.ru/cgp
(495) 508-99-59 / cgp@rusonyx.ru

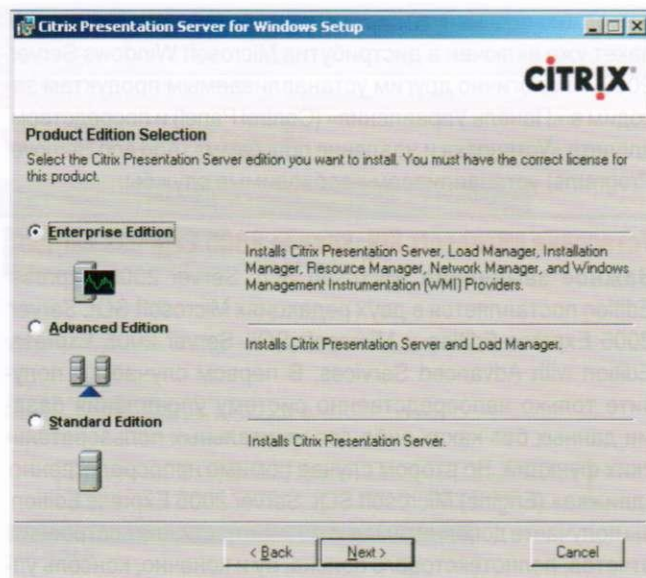


Рисунок 4. Окно выбора редакции Citrix Presentation Server 4.5

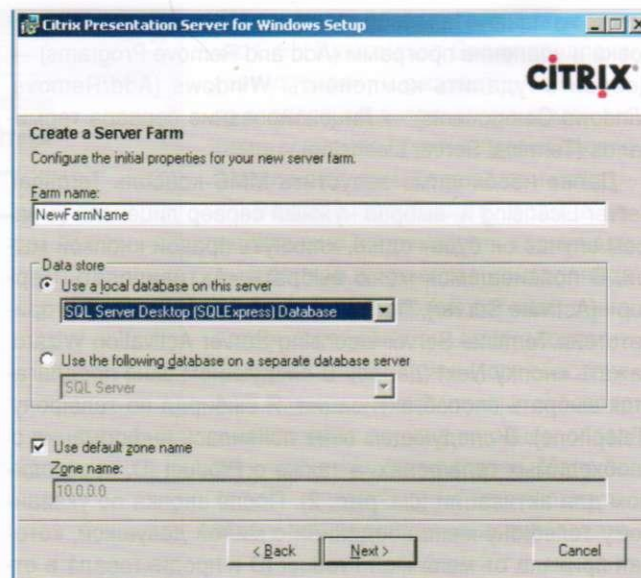


Рисунок 5. Выбор настроек новой фермы серверов

или именованный экземпляр. Разница заключается в том, что к экземпляру по умолчанию можно обращаться по имени сервера, на котором он установлен, а для обращения к базе данных именованного экземпляра нужно указать не только имя сервера, но и имя экземпляра. Обычно именованные экземпляры устанавливаются как второй, третий и т. д. на сервер, где уже установлен основной экземпляр по умолчанию, хотя бывают и исключения.

Для этого в коммерческих версиях MS SQL 2005 при установке программы-установщик предлагает выбрать имя экземпляра. Но в Microsoft SQL Server 2005 Express Edition при обычном способе установки такой выбор не предоставляется.

В то же время для нормальной установки Citrix Presentation Server с MSDE 2005 в качестве хранилища Data Store необходимо, чтобы Microsoft SQL Server 2005 Express Edition был установлен в качестве именованного экземпляра с именем CITRIX_METAFRAME. Как разрешить данное противоречие?

Можно решить проблему, что называется, «в лоб» и установить сначала один экземпляр по умолчанию MSDE 2005, а следом за ним начать установку второго экземпляра. В этом случае программа-установщик предложит выбрать другое имя устанавливаемого экземпляра, чем можно воспользоваться и задать требуемое имя CITRIX_METAFRAME. Но устанавливать лиш-

ний экземпляр приложения, довольно требовательного к ресурсам, абсолютно не хочется. Должен быть другой выход.

И этот выход был найден. Дистрибутив загружается с сайта Microsoft в сжатом виде. При обычной установке в качестве экземпляра по умолчанию достаточно запустить загруженный файл SQLEXPRESS.EXE или SQLEXPRESS_ADV.EXE. Для использования параметров, отличных от установки по умолчанию, нам необходимо запустить самораспаковывающийся архив с ключом -x:

```
SQLEXPRESS.EXE -x
```

или

```
SQLEXPRESS_ADV.EXE -x
```

в зависимости от используемой версии. Появится маленькое окошко с предложением ввести имя каталога для распаковки. Лучше указать дополнительную поддиректорию, иначе программа распаковки вывалит содержимое архива в тот же каталог, где лежит скачанный архив.

Перейдя в каталог распаковки, находим файл template.ini, из которого становится ясно, что для установки Microsoft SQL Server 2005 Express Edition в качестве именованного экземпляра нужно запустить установку с ключом:

```
INSTANCENAME=ИМЯ_ЭКЗЕМПЛЯРА
```

в нашем случае CITRIX_METAFRAME. Запускаем:

```
setup INSTANCENAME=CITRIX_METAFRAME
```

Далее установка ничем не отличается от установки MS SQL Server 2005.

Установка Citrix Presentation Server

Лучше всего устанавливать Citrix Presentation Server при помощи стандартной консоли установки. Обычно она стартует в режиме автозапуска (Autorun) при помещении установочного диска в CD-ROM. При запуске появится окно с меню из следующих пунктов:

- View installation checklist;
- Product installations and Updates;
- Citrix on the Web;
- Install Document Center.

Перед началом установки настоятельно рекомендуется еще один раз просмотреть checklist и выполнить подготовительные пункты, которые были пропущены. Если все готово и вы в этом уверены, приступаем к пункту Product installations and Updates.

Перед нами откроется новое меню. Необходимо выполнить все пункты по порядку.

Установка сервера лицензий

Так как переназначение дисков (Remap drivers) мы выполнили сразу после установки операционной системы,

то сразу переходим ко второму пункту – установка.

Следующий шаг – необходимо установить сервер лицензий. Выбираем Product installation and updates в окне установки Citrix, в следующем окне Install Citrix Licensing. Во время установки сервера лицензий ничего особенного не происходит. В одном окне вам предложат ознакомиться с лицензионным соглашением, потом выбрать компоненты инсталляции: License Server и/или License Management Console, далее предложат указать путь для установки, потом выбрать тип установленного веб-сервера: IIS или Apache и в конце концов предложат на перезагрузку IIS – установить переключатель OK to restart Microsoft IIS Server. После этого в меню «Пуск → Программы» («Start → Programs») появится подменю «Citrix → Management Console» (см. рис. 3).

Из данной консоли вы можете скачать с сайта Citrix.com и установить дополнительные файлы лицензий.

Установка Citrix Presentation Server

После установки License Server приступаем к установке непосредственно самого Presentation Server 4.5.

В окне установки выбираем Product installation and updates Citrix, в следующем окне Install Citrix Presentation Server 4.5 and its components. И снова знакомимся с лицензионным соглашением, выбираем компоненты для инсталляции (лучше оставить список без изменений).

Далее нам предложат установить сервер лицензий, но поскольку его мы уже устанавливали, оставляем переключатель на I already have a license server, or will use the product CD to install one later.

После этого нам продемонстрируют полный список устанавливаемых компонентов. На данном этапе, сняв отметки, можно отказаться от установки тех или иных компонентов, но лучше этого не делать.

Далее попадаем непосредственно в окно Citrix Web Installation Setup. Единственный важный пункт здесь – выбор или отказ от установки.

Следующий этап – непосредственная установка Citrix Presentation Server 4.5. Здесь вам предложат

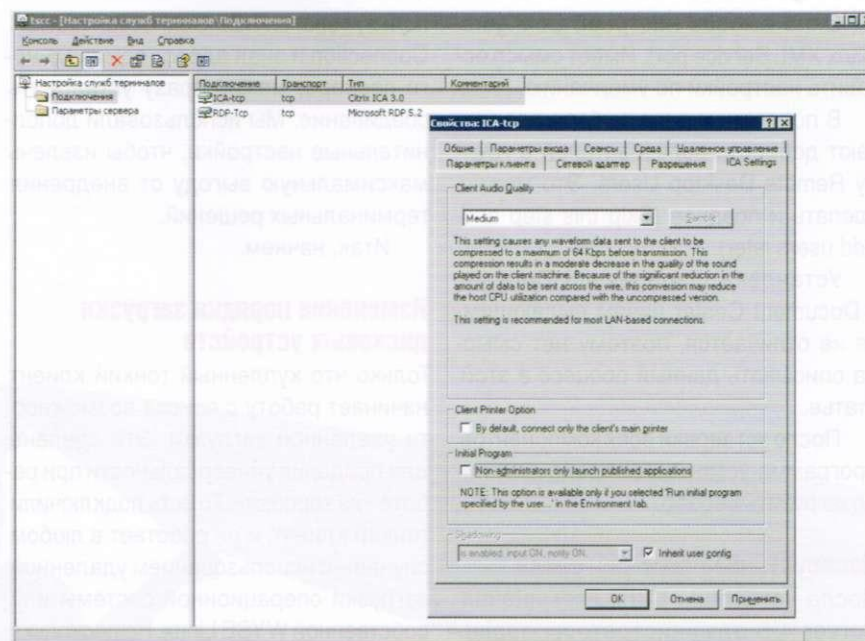


Рисунок 6. Разрешить вход на сервер рядовым пользователям

выбрать устанавливаемую редакцию продукта: Enterprise Edition, Advanced Edition или Standard Edition (см. рис. 4).

Данные редакции в первую очередь отличаются возможностями и соответственно стоимостью. За подробной информацией лучше всего обратиться на официальный сайт Citrix. Мы при покупке выбрали редакцию Enterprise Edition, поэтому оставили отмеченный первый пункт без изменений.

Последующие пункты инсталляции лучше оставить по умолчанию вплоть до Create or Join Server Farm. Так как мы создаем новую ферму, выбираем пункт Create a new farm.

В окне задания параметров для новой фермы вам предложат выбрать имя создаваемой фермы и самое главное – выбрать тип хранилища (Data store) для записи служебной информации. Выбираем пункт Use a local database on this server, и в качестве базы данных выбираем SQL Server Desktop (SQL Express) Database. Пункт Use default zone name я оставил без изменений (см. рис. 5).

Появится окно выбора Windows пользователя для администрирования фермы серверов. По умолчанию предлагается учетная запись пользователя, из-под которой производится установка.

В следующем окне предлагается разрешить IMA-шифрование. Поскольку повышенных требований к секрет-

ности не стоит, я оставил данный пункт без изменений.

В следующем окне предлагаю ввести доменное имя (имя хоста), где запущен License Server (Enter the host name of the machine hosting your Citrix License Server) или сделать это позднее (Enter the correct host name later). Поскольку и License Server и Presentation Server у нас установлены на одном сервере, я указал имя данного хоста.

Далее появится окно ввода Configure shadowing. Под теньным управлением (shadowing) в Citrix Presentation Server понимается возможность управления сеансами пользователя. В данном окне вы можете запретить (Prohibit shadowing of user session on this server) или разрешить использование (Allow shadowing of user session on this server).

В случае разрешения вы можете также:

- Запретить управление (Prohibit Remote Control), оставив только возможность наблюдения за действиями пользователя.
- Требовать подтверждения (Force a shadow acceptance popup). Эта опция запрещает теньные сеансы без подтверждения пользователей.
- Регистрировать все теньные соединения (Log all shadow connections). События установления теньных сеансов будут регистрироваться в журнале.

Окно выбора настроек Configure Citrix XML Service port. Имеет смысл оставить настройки по умолчанию.

В последнем окне выбора предлагают добавить пользователей в группу Remote Desktop Users. Это можно сделать и позднее (Skip this step and add users later).

Установка management Console и Document Center ничем выдающимся не отличается, поэтому нет смысла описывать данный процесс в этой статье.

После установки всех компонентов программа-установщик предложит перезагрузить сервер.

Послеустановочная настройка

После установки Citrix Presentation Server 4.5 администраторы сервера автоматически получают возможность открывать терминальную сессию. Другие пользователи, кроме администраторов, могут только запускать опубликованные приложения и не получают доступ к «Рабочему столу» Windows. Так как мы собирались использовать тонкие клиенты, необходимо предоставить такую возможность. Для этого необходимо открыть MMC-консоль «Настройка служб терминалов», правой кнопкой кликнуть на объекте ICA-tcp, из появившегося меню выбрать «Свойства». В появившемся окне перейти на вкладку ICA Settings и снять галочку с пункта Non-administrators only launch published applications (см. рис. 6).

Также необходимо добавить нужных пользователей в группу безопасности «Удаленные пользователи рабочего стола» (Remote Desktop users).

Также в целях безопасности настоятельно рекомендуется выключить Citrix ICA Toolbar и убрать с «Рабочего стола» Windows, а также из меню «Пуск» рядовых пользователей не нужные для повседневной работы меню, ярлыки и т. д.

Настройка тонких клиентов

По правде говоря, выбранные нами тонкие клиенты WYSE S10 не нуждаются в особой настройке. Их можно доставать из коробки и устанавливать на рабочие места. При работающем DHCP-сервере они автоматически получат IP-адреса. После этого, ис-

пользуя значок RDP Connection или ISA Connection и зная адрес терминального сервера, можно сразу установить соединение. Мы использовали дополнительные настройки, чтобы извлечь максимальную выгоду от внедрения терминальных решений.

Итак, начнем.

Изменение порядка загрузки дисковых устройств

Только что купленный тонкий клиент начинает работу с поиска возможности удаленной загрузки. Это сделано для придания универсальности при работе «из коробки». То есть подключили тонкий клиент, и он работает в любом случае — с использованием удаленной загрузки операционной системы или собственной WYSE Linux. Но поскольку удаленная загрузка у нас не использовалась, решено было эту функцию отключить. На сайте компании WYSE нашли следующую информацию:

- Подключить клавиатуру.
- По клавише войти в BIOS. Пароль на вход: Fireport.
- Сменить порядок загрузки (Boot Order Configuration)
- Сохранить настройки.

К сожалению, при помощи имеющихся у нас USB-клавиатур Genius K639 и Defender E KS-910B проделать данный трюк не удалось. Такое впечатление, что тонкий клиент загружается настолько быстро, что не успевает отработать прерывание от перечисленных выше клавиатур. Пришлось раздобыть «родную» клавиатуру WYSE KU-8933 (сами тонкие клиенты поставляются без клавиатуры) и с ней обойти все терминалы. Следует также заметить, что клавишу нужно нажимать одновременно с включением тонкого клиента, в противном случае не успеете войти в BIOS до начала загрузки.

Настройка статического IP-адреса

Вероятность того, что терминальные пользователи станут переносить свои терминалы, как ноутбуки, крайне маловероятна. Исходя из этого, для ускорения загрузки всем терминальным клиентам были присвоены статические адреса. В итоге скорость загрузки возросла от 2 до 5 секунд. Мелочь, а приятно.

Для настройки статического IP нужно кликнуть правой кнопкой мыши по пустому экрану и появившемуся меню выбрать пункт «System Setup → Network». В появившемся окне можно настроить параметры IP-адресации, а также DNS и WINS-серверы.

Настройка автоматического входа в терминальную сессию

Чтобы избавить пользователей от необходимости лишнего раз кликать по непонятным значкам, а также по возможности избавить их от необходимости изменять настройки тонкого клиента, был настроен автоматический вход в терминальную сессию. Теперь при включении тонкий клиент автоматически подключается к серверу по ICA протоколу, и пользователь попадает в привычное окно ввода пароля.

Для настройки автоматического входа на терминальный сервер по протоколу ICA необходимо правой кнопкой мыши щелкнуть по значку Default ICA.

В появившемся окне во вкладке Connection в поле Host Names вводим IP-адрес или имя хоста, в полях Login Username, Password, Domain Name вводим соответственно имя пользователя, пароль (если необходимо, чтобы пользователь входил на сервер без ввода пароля, это очень небезопасно) и имя домена. Во вкладке Options необходимо отметить опцию Auto-connect on start up. Также имеет смысл установить переключатель с Window mode на Full screen mode.

Для повышения уровня безопасности и быстродействия системы имеет также смысл отключить удаленное подключение всех локальных устройств (Autoconnect to local devices), таких как локальные диски (Disk), принтеры (Printer), USB и т. д.

Заключение

В отличие от парка персональных компьютеров терминальное решение на базе сервера терминалов и тонких клиентов является прогрессивным шагом в сторону именно корпоративной инфраструктуры, позволяющей снизить расходы, повысить производительность и безопасность системы. В то же время простота реализации и гибкость настройки позволяет внедрить данное решение без особых проблем.



ТЕЛЕКОММУНИКАЦИИ++



**УСЛУГИ МЕЖДУГОРОДНОЙ
И МЕЖДУНАРОДНОЙ СВЯЗИ
ДЛЯ ВАШЕГО ОФИСА**

Индивидуальные тарифные планы*
позволяют экономить при звонках
в другие города и страны

20-40%

www.ttk.ru

8 800 505 55 77

Реклама: Лицензии Минсвязи № 33684, 43407, 37942, 43408

* Индивидуальные тарифные планы формируются
с учетом объемов потребления и распределения трафика
по основным направлениям.

ТЕРРИТОРИЯ

ТАРИФ

КАЧЕСТВО

Новшества в Windows Server 2008: стек TCP/IP



Андрей Бирюков

В новой версии ОС Windows Server 2008 стек TCP/IP был полностью переработан. Рассмотрим, какие новшества появились в нем.

Windows Server 2008 содержит множество различных нововведений. Такие давно используемые и хорошо всем знакомые вещи, как реализация стека TCP/IP в новой операционной системе, также претерпела весьма существенные изменения.

Основные изменения и структура

Приведу список основных изменений в новом сетевом стеке. Прежде всего это одновременная поддержка как IPv4, так и IPv6. Шестую версию IP сейчас поддерживают практически все выпускаемые операционные системы, так что поддержка в Windows 2008 была вполне ожидаема. Новая технология поддержки IPv4 и IPv6 называется Dual-IP.

Другим нововведением является поддержка так называемой Strong Host Model (усиленной модели хоста). Что это такое, лучше объяснить на при-

мере. Когда на какой-либо узел приходит обычный unicast-пакет, сетевой интерфейс принимает его, если IP-адрес назначения совпадает с одним из адресов, который присвоен сетевым интерфейсам на данной машине. Такая обработка пакетов называется слабой моделью хоста (Weak Host Model). В случае усиленной модели будет обработан пакет только с IP-адресом, соответствующим адресу интерфейса, на который он пришел. Использование слабой модели может сделать не только конкретный узел, но и всю локальную сеть менее устойчивой к несанкционированным проникновениям, так как в случае, если на внешний интерфейс узла, выполняющего роль маршрутизатора, послать пакет с адресом из внутренней, то при неаккуратной конфигурации такой пакет вполне может достичь точки назначения. С использованием усиленной модели даже в случае неаккуратной настройки правил маршрутизации такого не случит-

ся. В текущих реализациях TCP/IP на Windows Server 2003 и Windows XP используется слабая модель. Более подробно с моделями узлов можно ознакомиться в статье [1].

Next Generation TCP/IP стек снижает нагрузку на драйвер Network Driver Interface Specification (NDIS) и сетевые адаптеры при обработке TCP и других видов трафика. Данный функционал весьма полезен при работе в сетях с высокой пропускной способностью. Также в новой версии стека появилась возможность обработки трафика на многопроцессорных машинах с распределением нагрузки. Для высокопроизводительных сетевых приложений, прежде всего различных видеотрансляций, многопроцессорная обработка также будет полезной.

Еще одним важным нововведением в стеке Next Generation является возможность автоматической настройки определенных компонентов стека (например, масштабируемость TCP

receive window и других) без ручного внесения изменений в конфигурацию. Тут следует отметить, что в UNIX-системах подобные функции существуют уже давно и существенно упрощают жизнь системным администраторам. Так что реализация данного функционала в Next Generation Stack является несомненным шагом вперед.

Схематически новую архитектуру можно изобразить с помощью рисунка. Поясню работу стека последовательно, снизу вверх. За отправку и получение пакетов на интерфейс отвечает драйвер NDIS. Затем данные передаются драйверу tcpip.sys.

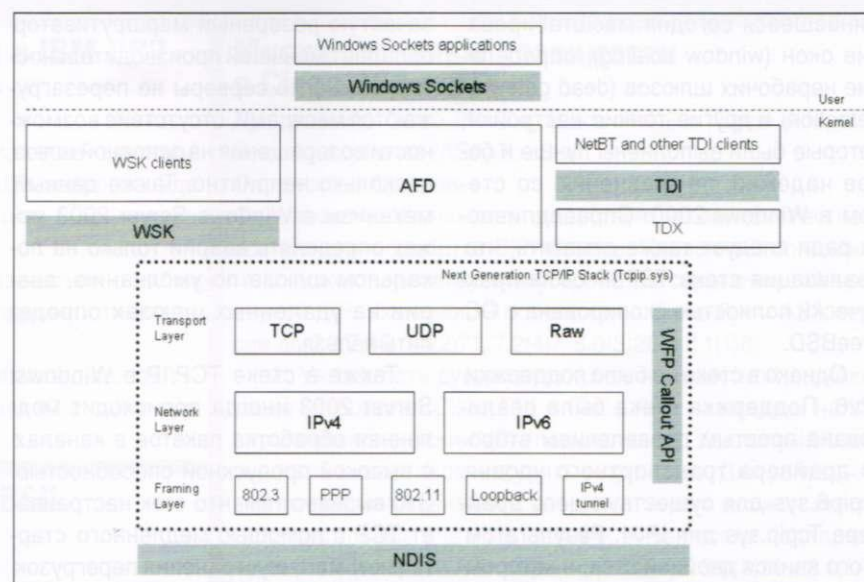
Архитектура драйвера новой версии стека TCP/IP, реализованного в файле tcpip.sys, состоит из следующих уровней:

- **Транспортный уровень** – содержит реализации протоколов TCP и UDP, а также механизм для отсылки базовых IP-пакетов, которым не требуется наличие TCP или UDP-заголовков.
- **Сетевой уровень** – содержит реализации протоколов IPv4 и IPv6 в виде уровня Dual IP layer (о данной технологии я уже писал выше).
- **Фреймовый уровень** – содержит модули для фрейминга пакетов IPv4 и IPv6. Существуют модули для интерфейсов IEEE 802.3 (Ethernet), IEEE 802.11 и Point-to-Point Protocol (PPP). Помимо этого существуют модули и для логических интерфейсов, таких как Loopback Interface, а также интерфейсов туннелирования на основе IPv4, которые часто используются в технологиях поддержки переключения с IPv4 на IPv6.

После такой обработки полученные данные передаются Winsock Kernel и другим обработчикам уровня ядра. И, наконец, в верхней части архитектуры находятся приложения Windows Socket, которые извлекают непосредственные данные и передают их приложениям.

Новые API

Также, рассказывая о новых функциях сетевого стека, нельзя не упомянуть о новых возможностях для программистов. Прежде всего это появление новых API для безопасности



Архитектура Next Generation TCP/IP

и фильтрации пакетов. Новый набор компонентов для программирования получил название Windows Filtering Platform (WFP). WFP обеспечивает возможность фильтрации на всех уровнях стека протоколов TCP/IP. Также следует отметить, что WFP более тесно интегрирован в реализацию TCP/IP для Windows Server 2008. Это позволяет сторонним разработчикам быстрее и легче разрабатывать сетевые драйверы, службы и приложения для работы с TCP/IP трафиком. Более подробно с Windows Filtering Platform можно ознакомиться в статье [2].

В продолжении темы API хотелось бы упомянуть три основных программных интерфейса, используемых приложениями, сервисами и другими системными компонентами для доступа к новому стеку TCP/IP – это WSK (Winsock Kernel), используемый клиентскими приложениями, Windows Sockets, используемый приложениями и сервисами (программный интерфейс Windows Sockets использует драйвер AFD – Ancillary Function Driver для выполнения функций на уровне сокетов через TCP/IP) и TDI (Transport Driver Interface), используемый NetBIOS over TCP/IP (NetBT) и другими устаревшими клиентами. Для трансляции запросов между TDI и новым стеком TCP/IP используется интерфейс TDX. По замыслу разработчиков WSK должен со временем заменить TDI.

Новый стек TCP/IP поддерживает программный интерфейс Callout API, который является унифицированным

способом для встраивания в стек и модификации данных на уровне пакетов. Программный интерфейс является частью WFP, обеспечивающей доступ к обработке пакетов на сетевом и транспортном уровнях. Новый стек также поддерживает отсылку и получение фреймов (frames), используя NDIS.

Завершая детальное описание нововведений, хочу привести краткий список функций, еще не упомянутых, однако заявленных разработчиком. Это улучшенная поддержка производительности и коррекция ошибок, поддержка аппаратных конфигураций и автонастройки, богатые функции расширяемости на уровне нового программного интерфейса, улучшенная поддержка рабочих станций мобильных пользователей, часто переключающихся из одной сети в другую, взаимодействие с Windows CE, Xbox и Windows Embedded, устойчивость к известным DoS и другим сетевым атакам.

Новые решения старых проблем

Теперь сравним архитектуру Next Generation TCP/IP Stack с предыдущими реализациями TCP/IP. Прежде всего вспомним некоторые особенности работы стека в предыдущей версии операционной системы и их недостатки. В пакетах обновлений (service packs) для Windows Server 2003 были включены дополнительные возможности: в TCP появилось уже упо-

минавшееся сегодня масштабирование окон (window scaling), определение нерабочих шлюзов (dead gateway detection) и другие тонкие настройки, которые были выполнены лучше и более надежно, по сравнению со стеком в Windows 2000. Справедливости ради следует также отметить, что реализация стека TCP/IP была практически полностью скопирована с ОС FreeBSD.

Однако в стеке не было поддержки IPv6. Поддержка стека была реализована простым добавлением второго драйвера транспортного уровня Tcpip.sys для существующего драйвера Tcpip.sys для IPv4. Результатом этого явился двойной стек, в котором транспорт для IPv4 и IPv6 поддерживался двумя различными драйверами, и для включения IPv6 администраторы должны были установить компонент протокола IPv6 на своих машинах с Windows XP и Windows Server 2003. Очевидно, что такой подход имеет несколько недостатков. Например, функциональность разделенного фильтра пакета для каждого протокола IP приводит к тому, что сложно настраивать брандмауэр на машинах, на которых работают оба протокола. Также это усложняет жизнь разработчикам сетевых приложений, которые необходимы для поддержки как IPv4, так и IPv6. А это означает дублирование кода и менее эффективный канал для выполнения стека.

Еще одним недостатком прежней реализации стека является механизм определения нерабочих шлюзов (dead gateway detection), который был введен как способ поддержания работоспособности, когда основной шлюз (gateway) или маршрутизатор (router) вышел из строя. В больших компаниях часто есть резервный маршрутизатор, который может работать в случае выхода из строя основного маршрутизатора, а определение нерабочих шлюзов позволяет использовать такие резервные шлюзы. Проблема реализации данного механизма в Windows Server 2003 заключалась в том, что не было возможности отката, другими словами, когда основной шлюз восстанавливал свою работоспособность, система продолжала отправлять пакеты на резервный шлюз до следующей перезагрузки. Так как

зачастую резервный маршрутизатор обладает меньшей производительностью, а многие серверы не перезагружаются месяцами, отсутствие возможности возвращения на основной шлюз несколько неприятно. Также данный механизм в Windows Server 2003 может определять аварии только на локальном шлюзе по умолчанию, аварии на удаленных шлюзах определить нельзя.

Также в стеке TCP/IP в Windows Server 2003 иногда происходит медленная обработка пакетов в каналах с высокой пропускной способностью. Это вызвано тем, что стек настраивает TCP с помощью медленного старта (slow start) и устранения перегрузок (congestion avoidance) — два стандартных алгоритма TCP/IP, которые были спроектированы еще до того, как гигабайтные сети стали нормой. Например, при использовании DFS (Distributed File System) вы можете иметь сервер вместо концентратора, который может копировать большое количество данных, проходящих по высокоскоростной WAN через множество маршрутизаторов. Результатом является то, что WAN-соединение перегружается, и копирование занимает больше времени, чем должно, иногда намного больше.

Завершая перечисление недостатков старого стека, хотелось бы упомянуть о проблемах с безопасностью. Дело в том, что одна таблица маршрутов (routing table) используется для разделения трафика для каждой из сетей, к которым подключен сервер, в том числе и VPN-соединения, что в свою очередь означает, что если у пользователя есть права локального администратора на машине, то он имеет возможность для изменения таблицы маршрутов таким образом, чтобы создать угрозу безопасности локальной сети, разрешив доступ из Интернета.

Теперь рассмотрим, каким образом приведенные выше недостатки были устранены в новой версии стека.

Проблема взаимодействия IPv4 и IPv6 решена с помощью уже упоминавшейся выше технологии Dual IP layer, которая существенно упрощает взаимодействие этих двух протоколов.

В Windows Server 2008 полностью решена проблема гарантированной

изоляции трафика VPN-соединением сессий входа, таблиц маршрута и сетевых интерфейсов в логическую конструкцию под названием подсистема выбора маршрута (routing compartment). Такое деление означает, что, например, соединение от ноутбука к Интернету и корпоративной сети с помощью VPN полностью изолированы одно от другого, что существенно увеличивает защищенность корпоративной сети.

Для решения проблемы отката в механизме определения нерабочих шлюзов (dead gateway detection) используется похожий набор методов, которые IPv6 использует для определения недоступных соседей в сетях IPv4, что в основном включает мониторинг сессий TCP и обмен пакетами ARP. Когда система обнаруживает такую ситуацию, стек переключается на следующий шлюз по умолчанию из списка, но стек продолжает мониторинг предыдущего шлюза, и если тот возобновляет работу, то система переключается обратно на этот основной шлюз. Это позволяет обеспечить оптимальную связь с удаленными офисами, гарантируя, что резервные шлюзы будут использоваться, только пока не работает основной шлюз.

В новом стеке проблема низкой производительности в сетях с высокой пропускной способностью решена с помощью нового алгоритма вместо алгоритмов, использовавшихся на других платформах. Этот новый подход называется Compound TCP (CTCP), и он может значительно повысить производительность при использовании в различных сетях.

Надеюсь, что информация о новшествах в стеке протокола TCP/IP будет вам полезна при работе с Windows Server 2008.

1. <http://technet.microsoft.com/en-us/magazine/cc137807.aspx> — описание Weak And Strong Host Models.
2. <http://www.microsoft.com/whdc/device/network/WFP.mspx> — описание Windows Filtering Platform.
3. [http://technet.microsoft.com/ru-ru/library/bb878108\(en-us\).aspx](http://technet.microsoft.com/ru-ru/library/bb878108(en-us).aspx) — статья на TechNet, посвященная Next Generation TCP/IP.
4. <http://www.netdocs.ru/articles/TCP-IP-Networking-Windows-Vista.html> — сеть TCP/IP в Windows Vista.

Множественные уязвимости в IBM DB2

Программа: IBM DB2 8 версии до Fixpak 17.

Опасность: Средняя.

Описание: 1. Уязвимость существует из-за неизвестной ошибки при обработке CONNECT и ATTACH-запросов. Удаленный пользователь может вызвать отказ в обслуживании приложения.

2. Уязвимость существует из-за неизвестной ошибки, относящейся к DB2FMP-процессам. Подробности уязвимости не разглашаются.

URL производителя: www-3.ibm.com/software/data/db2.

Решение: Установите исправление Fixpak 17 с сайта производителя.

Отказ в обслуживании в NetBSD

Программа: NetBSD 4.0.

Опасность: Средняя.

Описание: Уязвимость существует из-за ошибки деления на ноль при обработке ICMPv6-сообщений. Удаленный пользователь может с помощью специально сформированного ICMPv6 MLD-QUERY-пакета, в котором поле Maximum-Response-Delay установлено в значение, меньше чем 0x0010, вызвать отказ в обслуживании системы.

URL производителя: www.netbsd.org.

Решение: Установите исправление из CVS-репозитория производителя.

Отказ в обслуживании в FreeBSD

Программа: FreeBSD 6.x, 7.0.

Опасность: Средняя.

Описание: Уязвимость существует из-за ошибки в функции `icmp6_mtudisc_update()` в файле `src/sys/netinet6/icmp6.c` при обработке ICMPv6-сообщений «Packet Too Big». Удаленный пользователь может с помощью специально сформированного ICMPv6-сообщения вызвать панику IPv6-стека.

URL производителя: www.freebsd.org.

Решение: Установите исправление с сайта производителя.

Уязвимости при обработке CHM-файлов в ClamAV

Программа: ClamAV версии до 0.94.

Опасность: Средняя.

Описание: 1. Уязвимость существует из-за ошибки в файле `libclamav/chmunpack.c` при обработке CHM. Удаленный пользователь может с помощью специально сформированного CHM-файла заставить приложение обратиться к некорректному адресу в памяти и аварийно завершить свою работу.

2. Уязвимость существует из-за ошибки разыменования нулевого указателя в `libclamav`. Удаленный пользователь может вызвать отказ в обслуживании приложения.

3. Уязвимости существуют из-за утечки памяти в `freshclam/manager.c`. Удаленный пользователь может потратить все доступные ресурсы на системе.

4. Уязвимость существует из-за неизвестных ошибок, относящихся к `error path`, в `libclamav/others.c` и `libclamav/sis.c`.

URL производителя: www.clamav.net.

Решение: Установите последнюю версию 0.94 с сайта производителя.

Множественные уязвимости в Cisco ASA и PIX

Программа: Cisco PIX 7.x; Cisco PIX 8.x; Cisco Adaptive Security Appliance (ASA) 7.x, 8.x.

Опасность: Средняя.

Описание: 1. Уязвимость существует из-за неизвестной ошибки при обработке SIP-пакетов. Удаленный пользователь может перезагрузить устройство с включенной проверкой SIP. Уязвимости подвержены Cisco PIX и ASA версии до 7.0(7)16, 7.1(2)71, 7.2(4)7, 8.0(3)20 и 8.1(1)8.

2. Уязвимость существует из-за неизвестной ошибки, когда устройства Cisco PIX и Cisco ASA настроены на завершение клиентских VPN-подключений. Удаленный пользователь может перезагрузить устройство. Для успешной эксплуатации уязвимости злоумышленник должен иметь действительные учетные данные. Уязвимости подвержены Cisco PIX и Cisco ASA версии до 7.2(4)2, 8.0(3)14 и 8.1(1)4. Версии 7.0 и 7.1 не уязвимы.

3. Уязвимость существует из-за утечки памяти в Cisco ASA-устройствах, сконфигурированных на завершение безклиентных VPN-соединений. Удаленный пользователь может с помощью специально сформированного SSL или HTTP-пакета вызвать перезагрузку устройства. Уязвимости подвержены Cisco ASA версии до 7.2(4)2, 8.0(3)14 и 8.1(1)4. Версии 7.0 и 7.1 не уязвимы.

4. Уязвимость существует из-за ошибки при обработке URI в Cisco ASA-устройствах, которые завершают безклиентные удаленные VPN-соединения. Удаленный пользователь может вызвать перезагрузку устройства. Уязвимости подвержены Cisco ASA версии до 8.0(3)15, and 8.1(1)5. Версии 7.0, 7.1 и 7.2 не уязвимы.

5. Уязвимость существует из-за неизвестной ошибки в Cisco ASA-устройствах, сконфигурированных на завершение безклиентных VPN-соединений. Злоумышленник может обманом заставить пользователя посетить специально сформированный веб-сайт или ответить на e-mail и получить доступ к именам пользователей и паролям. Уязвимости подвержены Cisco ASA-устройства версии до 8.0 или 8.1 с включенной поддержкой безклиентных VPN. Версии 7.0, 7.1 и 7.2 не уязвимы.

URL производителя: www.cisco.com

Решение: Установите исправление с сайта производителя.

Множественные уязвимости в VMware ESX Server

Программа: VMware ESX Server 2.x.

Опасность: Средняя.

Описание: 1. Уязвимость существует из-за ошибки в библиотеке `libpng`. Удаленный пользователь может вызвать отказ в обслуживании.

2. Уязвимость существует из-за использования уязвимого кода библиотеки `freetype`. Удаленный пользователь может скомпрометировать целевую систему.

URL производителя: www.vmware.com/products/server/esx_features.html.

Решение: В настоящее время способов устранения уязвимости не существует.

Составил Александр Антипов

Используем check_relay в Sendmail для борьбы со спамом

Сергей Супрунов

Борьба со спамом вечна, как и сам спам. Способов защиты придумано множество, но ничего по-настоящему эффективного до сих пор не реализовано. Поэтому иногда приходится прибегать к «полумерам» типа фильтрации по именам хостов, «типичным» для спамеров. Такие почтовые серверы, как Postfix и Exim, предлагают сравнительно прозрачные способы фильтрации соединений. Владельцы же Sendmail вынуждены попотеть, чтобы раскрыть весь потенциал этого MTA.

Защита от спама на сервере провайдера по сравнению с «корпоративным» почтовым сервером имеет свои особенности.

Во-первых, здесь нет жёстких правил документооборота, позволяющих быть уверенным, что никто не ждёт писем в одной из японских кодировок или идущих с израильских доменов.

Во-вторых, среднестатистический абонент обычно не настолько владеет компьютером, чтобы эффективно осуществлять «обучение» различных систем контентной фильтрации, а сисадмину заниматься анализом карантина непоки (в том числе и по этическим соображениям). Всё это делает системы типа Spamassassin или DSPAM не очень эффективными либо требующими повышенного внимания со стороны администратора.

К тому же контентная фильтрация совсем не способствует экономии трафика. Более того, если значительная часть писем будет «рубиться» на сервере после загрузки и не попадёт к конечным пользователям, то этот трафик приходится рассматривать как прямые убытки компании (то есть компания «покупает» спам, но клиенту уже не «перепродаёт»).

Из методов же, позволяющих трафик экономить, более-менее безопасно можно использовать разве что грейстинг. Хотя в моей практике встречались клиенты, головные офисы которых оказывались не в состоянии обеспечить должную работу почтового сервера (а виноват, как обычно, провайдер, поскольку «в другие районы всё отправляется»).

Эффективность грейстинга сравнительно высока (на моём сервере он отсекал до 95% «мусора»), тем не менее есть и некоторые недостатки:

- его не так уж сложно обойти;
- каждое соединение отнимает определённое время, поскольку решение принимается лишь после этапа «RCPT TO» SMTP-диалога (а то и после команды «DATA», если включены опции, призванные обеспечить взаимодействие с серверами, использующими конкурирующий алгоритм защиты от спама – callback);
- затрачиваются определённые ресурсы на ведение базы триплетов.

Про «фальшивые» имена

О блокировании соединений, разрешение имени для которых возвращает FORGED, следует сказать особо. Злоумышленники иногда используют следующий приём. Соединение устанавливается с некоторого IP-адреса, обратную зону для которого контролирует спамер (например, ему делегирована данная сеть адресов). Указать в PTR-записи можно всё что угодно, хоть barackobama.com. И поэтому имеет смысл дополнительно запросить IP-адрес хоста barackobama.com (на этот запрос уже должен ответить DNS-сервер, обслуживающий данный домен) и сравнить полученный IP-адрес с исходным. Правда, как показывает практика, в наше время под правило FORGED чаще попадают хосты, для которых по возвращённому в PTR-записи имени вообще невозможно определить IP-адрес (т.е. отсутствует соответствующая A-запись в прямой зоне):

Однако иногда можно встретить замечания, что подобная блокировка может привести к проблемам в случае «парковки» на один IP-адрес нескольких доменов. Но поскольку сервер сначала запрашивает PTR-

запись, а затем по полученному имени хоста – IP-адрес, то проблем быть не должно – в итоге при любом «раскладе» мы выйдем на единственный, исходный IP-адрес. Несоответствие же теоретически может возникнуть лишь в случае, когда возвращённое имя хоста указывает на несколько IP-адресов. На практике такое встречается крайне редко, когда используется балансировка нагрузки (показаны только интересующие нас строки):

Но и в этом случае Sendmail, похоже, правильно отрабатывает ситуацию, проверяя, чтобы исходный IP-адрес соответствовал хотя

бы одному из адресов (если их несколько), возвращённых функцией sm_gethostbyname (см. src/daemon.c). Так что есть все основания надеяться, что правило FORGED будет срабатывать только в случае действительных проблем с настройками DNS-сервера отправителя. Впрочем, если у вас остались сомнения, правило FORGED можно совсем удалить.

К тому же используемый мною milter-greylist (в связке с Sendmail) под нагрузкой иногда переставал вовремя отвечать на запросы, что приводило к тому, что соединения ещё дольше оставались открытыми, увеличивая число одновременно работающих экземпляров sendmail до максимально возможного, нарушая тем самым обслуживание клиентов.

Когда ресурсы сервера оказались практически исчерпаны, было принято решение о дополнительной фильтрации на первой линии обороны, призванной снизить нагрузку на milter-greylist и используемый для проверки на вирусы ClamAV, пусть даже и ценой «разумного риска» ложных срабаты-

ваний. От «чёрных списков» на основе DNSBL я почти отказался – слишком уж часто какой-то из «клиентских» серверов куда-нибудь попадает, и приходится разбираться с тамошними администраторами и клиентами. Поэтому я решил залезть в дебри sendmail.cf и воспользоваться способностью Sendmail выполнять фильтрацию сообщений на основе регулярных выражений.

Sendmail позволяет «вмешаться» в процесс приёма сообщения на различных этапах (см. рисунок). Поскольку наша задача – отсеять вероятный спам как можно раньше, то интерес для нас представляет в первую очередь набор правил check_relay, получающий в качестве параметра выраже-

```
$ grep "dns rule 2" /var/log/maillog | tail -1
```

```
Sep  8 15:29:11 myhost sm-mta[32267]: m88BTATA032267:
ruleset=check_rcpt, arg1=<energo@mydom.ru>,
relay=18971199139.user.veloxzone.com.br [189.71.199.139]
(may be forged), reject=550 5.7.1 <energo@mydom.ru>... Relaying
denied (dns rule 2). IP name possibly forged [189.71.199.139]
```

```
$ nslookup 18971199139.user.veloxzone.com.br
```

```
Server:      10.161.193.5
Address:     10.161.193.5#53

** server can't find 18971199139.user.veloxzone.com.br: NXDOMAIN
```

```
$ dig -x 77.88.21.3
```

```
3.21.88.77.in-addr.arpa. 13943 IN PTR www.yandex.ru.
```

```
$ dig www.yandex.ru
```

```
www.yandex.ru. 5865 IN A 93.158.134.3
www.yandex.ru. 5865 IN A 77.88.21.3
```


Синтаксис правил sendmail.cf

Весь синтаксис этого файла мы рассматривать не будем (при желании вы всегда найдёте нужную информацию в Интернете). Ограничимся лишь правилами проверки. Каждое правило начинается с символа R (он обязательно должен быть первым в строке, без ведущих пробелов). Далее следует «левая часть» (шаблон, напоминающий регулярное выражение) и через символ табуляции (не пробелы!) — «правая часть», то есть преобразование, вызывающее, что нужно сделать со входными данными, соответствующими шаблону. В шаблоне могут использоваться следующие лексемы:

- ✓ **\$*** — ноль и более токенов;
- ✓ **\$+** — один и более токенов;
- ✓ **\$-** — ровно один токен;
- ✓ **\$@** — ни одного токена;
- ✓ **\$=X** — соответствует некоторому элементу класса X;
- ✓ **\$~X** — не соответствует ни одному элементу из класса X.

Токен — это подстрока, разделённая пробелом или одним из символов, перечисленных в опции OperatorChars. По умолчанию это «.:_%@!^/[]+». Поэтому не рассматривайте шаблон как чистое регулярное выражение — они работают с отдельными символами, а «левая часть» правил в sendmail.cf — с токенами. То есть строка «qwe_asd» будет соответствовать регулярному выражению «.*_*», но не шаблону «\$*_»\$, поскольку символ «_» не является разделителем токенов, и, следовательно, входная строка представляет собой один-единственный токен.

Преобразование можно рассматривать как некоторую процедуру, которая, получая на входе строку, совпавшую с шаблоном, выполняет над ней некоторые дейст-

вия и возвращает результат. Типы возврата могут быть следующими (указывается в начале «правой части»):

- ✓ **\$:** — перейти на следующее правило;
- ✓ **\$@** — выйти из данного набора правил (вернуться к точке вызова набора);
- ✓ **\$#** — прекратить дальнейшую проверку правил.

Возврат \$# OK или \$@ OK (в зависимости от того, требуется ли проверка по другим наборам правил) означает «принять данное сообщение». Если возвращается \$#error, то соединение разрывается с выдачей следующего за \$#error сообщения (оно же записывается и в лог-файл). Кроме того, можно задать преобразование входящих данных, результат которого будет передан дальше. С помощью лексем вида \$1, \$2 и т. д. в правой части можно ссылаться на фрагменты шаблона (соответствующие каждой отдельной лексеме). Например, так можно преобразовать адрес вида domain.ru!user в user@domain.ru:

```
R$+!$+          $: $2@$1
```

Также в правой части часто используются следующие лексемы:

- ✓ **>ruleset** — вызов указанного набора правил;
- ✓ **\$&{macro}** — выполнение указанного макроса;
- ✓ **\$db args \$** — проверка аргументов args по карте db.

Правила исполняются последовательно (за исключением выходов из набора по \$@ и \$#), каждое следующее используется в качестве входных данных выход предыдущего правила. Если входные данные шаблону не соответствуют, они проходят дальше без преобразований.

ние вида «<hostname>\$!<ip-address>», где hostname — имя хоста, ip-address — IP-адрес источника соединения. (Наборы check_mail и check_rcpt срабатывают после SMTP-команд «MAIL FROM» и «RCPT TO», получая в качестве параметра соответственно адрес отправителя или адрес получателя. Также опцией FEATURE('compat_check') в mc-файле можно подключить не показанный на рисунке «общий» набор check_compat, принимающий в качестве параметра и адрес отправителя, и адрес получателя и выполня-

ющийся после приёма всего сообщения. Сейчас эти наборы нам не столь интересны.) По умолчанию check_relay (точнее, Basic_check_relay) отвечает за «прогон» соединений по базе access и за работу «чёрных списков» (dnsbl) и выполняется до того, как в работу включатся внешние фильтры (работающие через milter-интерфейс). Определять свои правила обработки лучше всего в наборе правил Local_check_relay (по умолчанию наборы Local_check_* пусты), который будет вызываться первым, до

Basic_check_relay. Синтаксис рассмотрим ниже, после того как определимся с общими правилами фильтрации.

Определяем правила фильтрации

В большинстве случаев источником спама являются «пользовательские» подключения к сети Интернет, зачастую с динамическими IP-адресами. Это могут быть как машины самих спамеров, так и (что происходит гораздо чаще) заражённые какой-нибудь гадостью компьютеры ничего не подозревающих простых абонентов. Так что наша задача — определить правила, позволяющие предположить, является ли клиент «законным» почтовым сервером, или это обычный абонентский компьютер, с которого прямая отправка почты не предусматривается (скорей бы уж SPF начал приносить реальную пользу...). Попытаемся найти наиболее характерные отличия в именах «правильных» и «неправильных» хостов.

Но сначала ненадолго отойдём от собственно имён, чтобы рассмотреть другой вопрос. В идеале каждый почтовый сервер должен иметь доменное имя, тем более что таково требование RFC2821 (хотя теоретически можно отправлять почту и на user@[1.2.4.5], но на практике с таким сталкиваться не приходилось), в то время как адреса, динамически назначаемые клиентам, зачастую в DNS не представлены. Конечно, имя почтового домена и доменное имя SMTP-сервера — вещи несколько различные, и к имени сервера особых требований не предъявляется, но всё же серверы без PTR-записи мне пока не встречались. Так что хотя вероятность ложных срабатываний и остаётся, но ею, на мой взгляд, можно пренебречь, поэтому первым правилом введём «Блокировать почту с адресов, не имеющих PTR-записи». Здесь нам поможет не регулярное выражение, а способность Sendmail выполнять DNS-разрешение клиентского адреса и осуществлять то или иное действие в зависимости от результата (см. ниже листинг конфигурации, четыре строки, начиная с обращения к макросу &{client_resolve}). В данном примере я показал блокировку всех трёх возможных ошибок — TEMP, FORGED и FAIL, то есть помимо отсутствия PTR-

записи (FAIL) блокируются также временные ошибки DNS и ситуации, когда запрос A-записи по доменному имени возвращает IP-адрес, отличный от исходного (FORGED).

Кстати говоря, проверку на разрешение имени в последних версиях Sendmail можно включить и просто опцией FEATURE('require_rdns'). Но настройка вручную позволит лучше контролировать как порядок, так и состав проверок.

Как показывает практика, львиная доля спама идёт с адресов вида 123-45-67-89-nekiy.domen.provaidera.ru или m123.dialup.domain.com. Именно такие имена хостов провайдеры, не мудрствуя лукаво, обычно назначают динамическим соединениям. То есть либо в имени присутствует в том или ином виде сам IP-адрес (цифры, разделённые точками, подчёркиваниями, дефисами и иногда символами «х»), либо фигурируют некоторые характерные «слова» — dial, ppp, adsl, dynamic, dhcp, pool, client и т. п. Оформим эти случаи в следующие правила (поскольку на второй линии обороны у нас стоит milter-greylist, то правила будут достаточно строгими, чтобы свести к минимуму риск блокировки нормальных соединений).

Правило 1: Имя хоста содержит четыре группы цифр, разделённых символами [._x-]

Соответствующее регулярное выражение будет выглядеть так: ([0-9]{1,3}[._x-]){4}. Под это выражение будут подпадать имена следующего вида (показанные здесь и далее имена хостов взяты из лог-файла и соответствуют хостам, с которых реально поступал спам; если кто увидит свой домен — я не специально):

- cpe-98-27-181-209.neo.res.rr.com;
- dialup-88-147-133-182.san.ru;
- 109.251-224-87.telenet.ru;
- 125x103x16x176.ap125.ftth.ucom.ne.jp;
- 219-180-207-85.zapcechy.adsl-llu.static.bluetone.cz.

К сожалению, нередко имена, включающие меньшее число цифровых групп:

- 100-240-90.adsl.terra.cl;
- 103-185.trifle.net.

Можно, конечно, в предыдущем правиле просто уменьшить число ожидаемых групп до двух, но по соображениям удобства отладки (учитывая, что вероятность ложных срабатываний по данному правилу несколько выше) выделим его отдельно.

Правило 2: Имя хоста содержит как минимум две группы цифр

В порядке ужесточения также потребуем, чтобы это не было частью имени домена 2-го уровня (но при этом уберём

Тестирование правил

Как известно, от ошибок никто не застрахован. Синтаксис регулярных выражений, если можно так сказать, «поощряет» опечатки — один пропущенный символ экранирования перед точкой, и результат может уже довольно сильно отличаться от ожидаемого. Поэтому каждое правило имеет смысл предварительно протестировать (тем более, что разрабатывать правила вы, скорее всего, будете, ориентируясь на вполне определённые имена хостов). Сделать это можно следующим образом:

```
$ sendmail -bt
```

```
ADDRESS TEST MODE
(ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
```

```
> /map block4 pool-127-165.gen.ua
```

```
map_lookup: block4 (pool-127-165.gen.ua)
no match (0)
```

```
> /map block5 pool-127-165.gen.ua
```

```
map_lookup: block5 (pool-127-165.gen.ua)
returns @MATCH (0)
```

```
> /quit
```

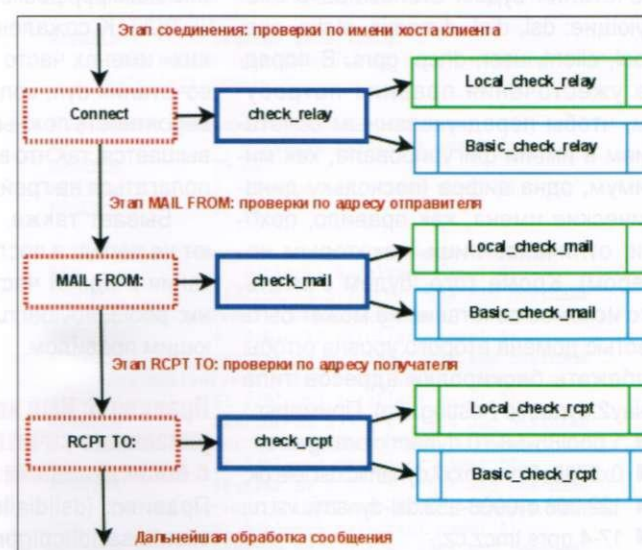
Команда /map проверяет имя хоста на соответствие той или иной «карте». Наши правила тоже являются картами. Если соответствие найдено, возвращается @MATCH, если нет — вы получите сообщение «no match».

Проводить тестирование лучше всего после компиляции нового конфигурационного файла (make), но до его инсталляции (make install) и перезапуска работающего почтового сервера (make restart). В этом случае потребуются явно указать новый конфигурационный файл (sendmail -bt -C mydom.ru.cf), но зато в случае ошибок вы сможете их исправить до того, как они скажутся на реальной работе.

верхнюю границу на число цифр в группе, так как цифры в данном случае не всегда означают часть IP-адреса): ([0-9]+[._x-]){2,}.*(\.){2,}. Продолжим.

Правило 3: Имя хоста содержит более восьми шестнадцатеричных цифр подряд

Иногда встречаются и такие имена, как bd21a1bd.virtua.com.br (обычно цифр 8, т.е. записывается IP-адрес в hex-формате, но может встречаться и больше). Поскольку здесь участвуют и буквы, то можно «нарваться» на добропорядочный адрес типа affeccade.cho-to.tld, так что дополнительно потребуем, чтобы такая группа символов была частью домена не менее четвёртого уровня: [0-9a-f]{8,}(\.){3,}. Ну и для себя отметим, что здесь требуется дополнительное внимание к логам (по-хорошему, это правило лучше бы вообще исключить, но мне пока оно проблем не создавало; вас же я предупредил).



Порядок вызова наборов правил check_*

Что ответить на HELO?

Наверняка вы обратили внимание, что в показанной схеме один этап отсутствует – команда HELO/EHLO, которой клиент (под «клиентом» здесь и далее подразумевается программа, устанавливающая соединение с вашим сервером) приветствует сервер сразу после подключения. Согласно стандарту в HELO/EHLO клиент должен передать своё полное имя (FQDN), которое, очевидно, должно соответствовать IP-адресу клиента. Однако на практике здесь может быть множество «подводных камней».

Во-первых, у различных клиентов могут быть различные понятия о стандартах (особенно это касается собственно почтовых клиентов типа Outlook, The Bat! и т. п., так что в первую очередь фильтрация по HELO может сказаться на ваших же пользователях) – не всегда в HELO будет указано именно то, что нужно. Во-вторых, клиент может соединяться через какой-нибудь NAT-или прокси-сервер и, передавая HELO, может понятия не иметь, с какого IP-адреса фактически будет идти соединение. Наконец, в-третьих, у клиента может быть несколько интерфейсов, и не факт, что FQDN будет соответствовать именно тому, с которого осуществляется подключение к серверу. В общем, на мой взгляд, «безусловная» фильтрация по HELO слишком опасна, и её лучше исключить.

Но, с другой стороны, такая проверка может быть весьма эффективной. Так что если вы всё-таки хотите «наказать» кли-

ентов, не соблюдающих стандарт (кстати говоря, в RFC 2821 ясно обозначено, что сервер не должен отказывать в соединении на основании неправильного имени в HELO/EHLO), можете воспользоваться рядом приёмов, которые можно найти в Интернете. Обычно ограничиваются блокировкой клиентов, которые сообщают в приветствии имя сервера, к которому подключаются (такая блокировка сравнительно безопасна, но тоже могут быть нюансы). Вот несколько ссылок:

- ☑ <http://www.opennet.ru/openforum/vsluhforum/D1/57895.html>;
- ☑ <http://www.mta.org.ua/sendmail/stuff/www.cs.niu.edu/~rickert/cf/bad-ehlo.html>;
- ☑ http://www.mta.org.ua/sendmail/stuff/www.cs.niu.edu/~rickert/cf/hack/block_bad_helo.m4;
- ☑ <http://www.vttoh.com/heloehlo.htm>.

Наконец, можно (целиком положившись на разработчиков Sendmail) добавить в mc-файл опцию FEATURE('block_bad_helo'). Любопытных отсылаю к файлу cf/m4/proto.m4, начиная со строки «ifdef('_BLOCK_BAD_HELO_', 'dnl», где можно посмотреть, какие именно правила будут применяться.

Однако, на мой взгляд, проверку по HELO лучше использовать исключительно в комплексных системах типа Spamassassin как один из критериев «спамности». Но эта тема уже далеко выходит за рамки статьи.

Правило 4: Имя хоста содержит одно из указанных сочетаний

Сочетания будем отслеживать следующие: dsl, dial, dynamic, static, ppp, pool, client, user, dhcp, gprs. В порядке ужесточения правила потребуем, чтобы перед указанным сочетанием в имени фигурировала, как минимум, одна цифра (поскольку динамические имена, как правило, похожи, отличаясь лишь некоторым номером). Кроме того, будем считать, что искомое сочетание не может быть частью домена второго уровня (чтобы избежать блокировки адресов типа relay2.dynamic-hosting.org). Примеры:

- 1.pool85-58-70.dynamic.orange.es;
- 0x5735c879.esnxx7.dynamic.dsl.tele.dk;
- 122.208.c10008-a53.dsl-dynamic.vsi.ru;
- 17-4.gprs.tmcz.cz;
- abfx26.neoplus.adsl.tpnet.pl.

Регулярное выражение получается довольно длинным: [0-9].*(dsl|dial|dynamic|static|ppp|pool|client|user|dhcp|gprs).*(\..*){2,}. К сожалению, в «динамических» именах часто попадаются также сочетания dyn, vpn и ip, но здесь уже вероятность ложных срабатываний повышется, так что в этом случае будем полагаться на грейлистинг.

Бывает также, что цифры следуют не перед, а после указанного сочетания в одной части доменного имени: pool3.provider.ru. Учтём это следующим правилом.

Правило 5: Имя хоста содержит указанные сочетания с последующими цифрами

Правило: (dsl|dial|dyn|static|ppp|pool|client|user|dhcp|gprs|ip|vpn|nat)[a-z_]*[0-9]+.*(\..*){2,}. Сочетания dyn, ip, vpn,

nat добавим сюда же, поскольку в данном контексте они уже более «безопасны».

И ещё одно правило предусмотрим для явного указания конкретных доменов, почту от которых мы не хотели бы принимать ни при каких обстоятельствах. Пример можно посмотреть ниже, в листинге конфигурации.

Правила-исключения

Пока будем считать, что этого достаточно. (Думаю, вы понимаете, что дальше вам предстоит не один месяц кропотливого анализа лог-файлов на предмет выявления ложных срабатываний и пропуска спама с последующей корректировкой данных правил.) Но нужно предусмотреть ещё кое-что.

Во-первых, рано или поздно вы столкнётесь с тем, что некоторые «хорошие» почтовые серверы попадут под ваши правила, а корректировка самих правил с целью учесть эту ошибку либо сделает правила излишне громоздкими, либо приведёт к увеличению числа спама, просачивающегося сквозь фильтрацию.

Во-вторых, если своим собственным клиентам вы назначаете DNS-имена так, что они попадают под одно из правил (либо не назначаете имён вообще), то может получиться так, что они не смогут отправлять почту через ваш сервер. Таким образом, мы приходим к необходимости создать правила-исключения, которые будут стоять до рассмотренных выше правил блокировки и пропускать те соединения, которые, безусловно, должны быть пропущены.

Таких правил для начала будет три:

■ `^(mailrelay|mx|smtp|mta).*[0-9]*(\..*){2,}` – то есть будем пропускать адреса вида relay2.mydom.com. (Сейчас они и без того будут проходить через наши правила без проблем, но, во-первых, лучше обработать потенциально частые соединения пораньше, чтобы не прогонять каждое через все правила, а, во-вторых, это позволит уменьшить тяжесть последствий ошибок в будущем, когда нам захочется «закрутить гайки».)

■ `\.(mydom\.rulrambler\.rulyandex\.rulmail\.ru)$` – это правило для доменов-исключений. Сюда имеет смысл занести свой домен, а также домены, с которых получить спам –

меньшее зло, нежели отвергнуть нужное сообщение. Рассматриваются лишь поддомены и отдельные хосты указанных доменов (предполагается, что сами доменные имена ни под одно из последующих правил блокировки не попадают). Не забывайте экранировать символы «.».

■ 10\5\{([0-9]){1,3}\.([0-9]){1,3} — здесь мы будем пропускать соединения с IP-адресов вида 10.5.x.y, предполагая, что данные адреса принадлежат нашим клиентам, для которых имена хостов в DNS не прописаны (а значит, в качестве таковых будут передаваться IP-адреса). Если «своих» сетей несколько, лучше для каждой из них указать отдельное правило.

Изменяем конфигурацию

Теперь добавим всё это в файл конфигурации Sendmail. Редактировать непосредственно sendmail.cf — не самая лучшая идея, так как этим вы обрежете себя на необходимость выполнять абсолютно всю настройку именно в этом файле, поскольку при использовании mc-файла внесённые в cf-файл изменения будут теряться. К счастью, «cf-фрагменты» при необходимости можно добавлять и в mc-файл, где они будут учитываться препроцессором m4 и переноситься в итоговый cf-файл.

Для этого в самом конце mc-файла (после опций MAILER) добавим следующее (длинные строки для удобства разбиты на несколько, строка-продолжение начинается с пробелов):

```
LOCAL_CONFIG
# Правила-исключения
Relays      regex -a@MATCH ^([mail|relay|mx|
smtp|mta]).*[0-9]*(\.)*{2,}
Randoms     regex -a@MATCH \. (mydom\.ru|hotmail\.com|
rambler\.ru|yandex\.ru|mail\.ru)$
Ournet1     regex -a@MATCH 10\5\{([0-9]){1,3}\.([0-9]){1,3}

# Правила блокировки
Hblock1     regex -a@MATCH ([0-9]){1,3}[\. x-]){4}
Hblock2     regex -a@MATCH ([0-9]+[\. x-]){2,}.*(\.)*{2,}
Hblock3     regex -a@MATCH [0-9a-f]{8,}(\.)*{3,}
Hblock4     regex -a@MATCH [0-9].*(dsl|dial|dynamic|
static|ppp|pool|client|user|dhcp|gprs).*(\..)*{2,}
Hblock5     regex -a@MATCH (dsl|dial|dyn|static|
ppp|pool|client|user|dhcp|gprs|ip|
vpn|nat)[a-z_-]*[0-9]+.*(\.)*{2,}
Hblock6     regex -a@MATCH (vnetranet\.pl|
mediating\.barrier\.volia\.net)$

LOCAL_RULESETS
$Local_check_relay

# Пропускаем соединения со своей подсети
R$*         $: $(ournet1 ${client_name}) $)
R$MATCH     $@ OK
```

Стандарт есть стандарт

Согласно RFC822 и RFC1123 каждый почтовый сервер должен безоговорочно принимать почту на адрес postmaster@<domain.tld>. RFC2142 добавляет к этому списку также адрес abuse@<domain.tld>. Это обусловлено тем, что у отправителя всегда должна быть возможность сообщить вам о проблеме. В нашем примере это требование стандарта не выполняется, т.к. в случае «неправильного» имени хоста наш сервер сразу разорвёт соединение, не спросив даже имени отправителя (MAIL FROM), не говоря уже о том, чтобы узнать, кому тот хотел написать письмо (RCPT TO).

Как было показано, проверки check_relay выполняются до проверок check_mail и check_rcpt. Однако этот порядок можно изменить. Для этого в mc-файл нужно добавить следующую директиву:

```
FEATURE('delay_checks', 'friend')
```

Это, во-первых, отложит проверки check_relay и check_mail до того момента, пока не пройдёт check_rcpt. Во-вторых, аргумент «friend», переданный директиве, позволяет задать в базе access правило SPAMFRIEND для нужных адресов:

```
To:postmaster@      SPAMFRIEND
To:abuse@            SPAMFRIEND
```

Пересобрав базу access (командой make в каталоге /etc/mail), вы обеспечите безоговорочное прохождение сообщений на указанные адреса, и лишь после

этого будут выполняться наши проверки check_relay. Кстати, в лог-файл информация о блокировках отныне будет заноситься с пометкой «ruleset=check_rcpt», поскольку теперь наборов правил check_relay и check_mail в sendmail.cf нет — вместо них при использовании вышеуказанной директивы появляются «нестандартные» checkrelay и checkmail соответственно, которые вызываются из check_rcpt. Побочным эффектом этого будет то, что в daily-сообщениях везде будет фигурировать ваш домен:

```
Checking for rejected mail hosts:
8252 mydom.ru (554... 1))
1492 mydom.ru (554... 2))
668 mydom.ru (554... 3))
534 mydom.ru (554... 4))
424 mydom.ru (554... 5))
111 mydom.ru (550... [218.9.79.68])
108 mydom.ru (550... [89.38.12.29])
91 mydom.ru (550... dul.ru)
76 mydom.ru (451... resolve)
```

Происходит это из-за того, что скрипт /etc/periodic/daily/460.status-mail-rejects (напомню, что речь идёт о FreeBSD) выбирает домен из подстроки arg1, записываемой в лог-файл для каждого факта блокировки, а для check_rcpt туда передаётся адрес получателя (т.е. адрес вашего пользователя). Кроме того, такое изменение порядка следования правил увеличивает нагрузку на систему. Но зато у отправителей будет больше шансов сообщить вам о допущенной вами ошибке.

```
# Выполняем разрешение имени клиента
R$*         $: < ${client_resolve} >
R<TEMP>      $error $@ 4.4.0 $: \
"450 Temp. (dns rule 1). Cannot resolve PTR for " \
${client_addr}
R<FORGED>     $error $@ 5.7.1 $: \
"550 Denied (dns rule 2). IP name forged " \
${client_name}
R<FAIL>       $error $@ 5.7.1 $: \
"550 Denied (dns rule 3). IP name lookup failed " \
${client_name}

# Применяем оставшиеся правила-исключения
R$*         $: $(relays ${client_name}) $)
R$MATCH     $@ OK
R$*         $: $(exdoms ${client_name}) $)
R$MATCH     $@ OK

# Блокируем соединения согласно правилам блокировки
R$*         $: $(block1 ${client_name}) $)
R$MATCH     $error $@ 5.7.1 $: "554 Bad hostname \
for mailserver: " ${client_name} " (regex rule 1)"
R$*         $: $(block2 ${client_name}) $)
R$MATCH     $error $@ 5.7.1 $: "554 Bad hostname \
for mailserver: " ${client_name} " (regex rule 2)"
R$*         $: $(block3 ${client_name}) $)
R$MATCH     $error $@ 5.7.1 $: "554 Bad hostname \
for mailserver: " ${client_name} " (regex rule 3)"
R$*         $: $(block4 ${client_name}) $)
R$MATCH     $error $@ 5.7.1 $: "554 Bad hostname \
for mailserver: " ${client_name} " (regex rule 4)"
R$*         $: $(block5 ${client_name}) $)
R$MATCH     $error $@ 5.7.1 $: "554 Bad hostname \
for mailserver: " ${client_name} " (regex rule 5)"
```



```
R$*      $: $(block6 ${client_name} $)
R$MATCH  $error $@ 5.7.1 $: "554 Bad hostname
for mailserver: " ${client_name} " (regex rule 6)"
```

Итак, что здесь происходит?

В секции LOCAL_CONFIG прописаны все наши правила. Строки, начинающиеся с «K» — это так называемые «карты» (maps), в нашем случае имеющие тип regex и проверяющие соответствие входящих аргументов указанному регулярному выражению.

Далее, в секции LOCAL_RULESETS, определяются локальные наборы правил преобразования. В нашем случае такой набор один — Local_check_relay. И здесь мы последовательно проверяем соответствие имени клиента (\${client_name}) нашим правилам.

Сперва безоговорочно пропускаем (\$@ OK) клиентов, подпадающих под правило ournet1. Затем выполняем разрешение имени клиента (\${client_resolve}) с выдачей соответствующих сообщений об ошибках в зависимости от результата. Если проверку на ournet1 делать после разрешения имени, то соединения наших клиентов, не имеющих PTR-записи, будут отклонены.

Клиенты, успешно прошедшие проверку на соответствие PTR- и A-записей, переходят к дальнейшим правилам. Сначала применяем наши правила-исключения, пропуская клиентов, имя которых соответствует данным регулярным выражениям (если у вашего домена есть проблемы с разрешением имён, то эти правила можно поставить сразу после ournet1; хотя лучше навести порядок в DNS).

Ну а затем блокируем соединения по нашим правилам блокировки. Для удобства последующего анализа лог-файла каждое сообщение об ошибке помечаем номером правила.

Запускаем и тестируем

Теперь можно ещё раз всё проверить (обратите особое внимание, чтобы левая и правая части каждой строки разделялись символами табуляции, а не пробелами), пересобрать sendmail.cf и поставить его на «боевое дежурство» (в данной статье я имею в виду FreeBSD; на вашей системе могут потребоваться другие команды):

```
# cd /etc/mail
# make && make install
# make restart
```

Результат работы можно будет получить, проанализировав лог-файлы (maillog.7.bz2 собран, когда данные правила фильтрации ещё не использовались; maillog.0.bz2 — уже с работающими регулярными выражениями):

```
$ cd /var/log
$ bzcat maillog.7.bz2 | grep "Greylisting in action" | wc -l
203944

$ bzcat maillog.7.bz2 | grep "stat=Sent" | wc -l
1188

$ bzcat maillog.7.bz2 | grep "5.7.1" | wc -l
1179

$ bzcat maillog.0.bz2 | grep "Greylisting in action" | wc -l
```

```
5760

$ bzcat maillog.0.bz2 | grep "5.7.1" | wc -l
44931

$ bzcat maillog.0.bz2 | grep "stat=Sent" | wc -l
1243
```

При желании можно получить статистику по конкретным сработавшим правилам, например, командой:

```
grep 'regex rule 3' maillog | wc -l
```

можно получить количество соединений, заблокированных третьим правилом. Поскольку делать подобный анализ нужно будет довольно часто, имеет смысл создать для этих целей shell-скрипт (один из возможных вариантов вы найдёте на сайте журнала, в разделе «Исходный код»):

```
$ ./getsendmailstat.sh /var/log/maillog.0.bz2

[Extracting /var/log/maillog.0.bz2 to /tmp/getsendmailstat...]
DNS rule 1:      704
DNS rule 2:     11913
DNS rule 3:     21472

Regex rule 1:    8248
Regex rule 2:    1492
Regex rule 3:     668
Regex rule 4:     534
Regex rule 5:     424
Regex rule 6:       0

Total 5.7.1:     44931
Greylisting:     5760
Total sent:      1243
[/tmp/getsendmailstat removed.]
```

Итак, вместо более чем 200 тысяч срабатываний greylisting за сутки, после включения правил фильтрации по имени хоста на долю milter-greylist осталось лишь 5-7 тысяч подключений. (При анализе этих данных нужно учитывать тот факт, что если сообщение отправляется сразу на несколько адресов получателей, то в случае greylisting создаётся запись в лог-файле для каждого адреса, блокировка же сопровождается одной записью на всё соединение. Так что можно считать, что одна строка «5.7.1» идёт за три-четыре «Greylisting in action».) Нагрузка на систему заметно снизилась, так что поставленную задачу можно считать решённой. Первое время придётся, конечно, критично просматривать лог-файлы и более чутко реагировать на жалобы клиентов, зато необходимость поиска нового «железа» для сервера ещё на некоторое время теряет свою актуальность.

1. Супрунов С. Greylisting: панацея от спама или «мыльный пузырь»? //Системный администратор, №7, 2006 г. — С. 12-15. — http://www.samag.ru/art/07.2006/07.2006_04.html.
2. Клаус Асман. Using check_* in sendmail 8.8 — <http://www.sendmail.org/~ca/email/check.html>.
3. Эрик Олман. Anti-Spam Configuration Control — http://www.sendmail.org/m4/anti_spam.html.
4. Д. Сени. Considerations for the use of DNS Reverse Mapping draft-ietf-dnsop-reverse-mapping-considerations-06 — <http://tools.ietf.org/html/draft-ietf-dnsop-reverse-mapping-considerations-06>.

Jabberd2 – простой и нетребовательный к ресурсам XMPP-сервер

Часть 2. Управление базами данных BerkeleyDB и тонкая настройка Jabberd2



Михаил Кондрин

В первой части статьи [1] была рассмотрена установка и базовая настройка XMPP-сервера Jabberd2. Однако дальнейшая настройка и отладка потребуют «хирургического» вмешательства и непосредственного редактирования его баз данных.

Итак, сервер запущен, список пользователей создан, с помощью Jabber-клиента Pidgin

можно обмениваться сообщениями как с пользователями нашего сервера, так и с внешними серверами. Что даль-

ше? Как вы помните, одним из условий техзадания для нашего Jabber-сервера был отказ от использования специ-

ализированных серверов баз данных для хранения учётных записей пользователей и другой необходимой информации. Поддержка сервера баз данных требует дополнительных усилий, так что гораздо проще хранить информацию в дисковых файлах, тем более что в большей степени эта информация представляет интерес только для самого сервера Jabberd2, и у вас вряд ли появится необходимость делиться ею с каким-нибудь другим сервисом. Также для доступа к этим дисковым файлам была выбрана библиотека BerkeleyDB.

Обычно базы данных BerkeleyDB характеризуют как не нуждающиеся в администрировании. Это справедливо в том смысле, что приложение может само решать административные вопросы в более-менее автоматическом режиме в процессе выполнения. В большинстве случаев это даже позволяет восстанавливать базы данных BerkeleyDB после аварийного завершения. Вопрос же создания резервных копий (вторая головная боль системных администраторов) при этом решается созданием специализированных утилит, которые решают задачу резервирования для всего приложения целиком.

Но это в том случае, если разработчики позаботились о создании таких инструментов. К сожалению, подход авторов Jabberd2 в том, что касается баз данных BerkeleyDB, не слишком конструктивен, поскольку ничего подобного они предложить не могут. Поэтому если вы решите хранить данные Jabberd2 средствами BerkeleyDB, то вам придётся волей-неволей вникать во внутреннее устройство и методы работы с базами данных этого типа.

Есть, правда, и хорошая новость — это не слишком сложно.

От BSD 4.4 до Oracle

История BerkeleyDB, библиотеки, которая в настоящее время насчитывает десятки миллионов инсталляций, началась в 1992 году, когда молодые сотрудники университета Беркли Марго Зельцер (Margo Seltzer) и Майк Олсон (Mike Olson) под руководством Кейта Бостича (Keith Bostic) взялись за модификацию утилиты ndbm (которая использовалась для работы с базами данных в оперативной памяти) к ре-

лизу операционной системы BSD 4.4. Эта модификация оказалась удачной, и уже в 1996 году Кейт Бостич и Марго Зельцер основали семейный бизнес (к тому времени они поженились) — компанию Sleepycat, которая занималась продвижением BerkeleyDB.

Благодаря сотрудничеству с университетом Мичигана (где был реализован один из первых серверов LDAP) и с компанией Netscape первоначальный проект был существенно усовершенствован и приобрёл широкую популярность.

В 2006 году компания SleepyCat (к которой к тому времени присоединился и Майк Олсон) была куплена лидером на рынке баз данных — компанией Oracle. До этого долгое время BerkeleyDB выходила под двойной лицензией, поэтому на программистских форумах слухи об этой покупке вызвали беспокойство, что Oracle закроет код BerkeleyDB. Однако эти опасения не подтвердились, и за последующие два года вышло несколько новых релизов этой библиотеки, причем Oracle продолжает выполнять свое обещание оставить код этой библиотеки открытым и свободным.

Все трое сотрудников SleepyCat по-прежнему работают над своим продуктом, совмещая программистскую деятельность с преподавательской: Марго Зельцер одновременно является профессором Гарвардского университета.

Администрирование баз данных BerkeleyDB

Как вы уже знаете, BerkeleyDB — это библиотека, которая позволяет нескольким приложениям совместно работать как над дисковыми, так и оперативными (расположенными в памяти) базами данных. Раз речь идёт о совместной работе с дисковым файлом, то возникает вопрос контроля доступа к этому файлу, чтобы информация для каждого из активных приложений была консистентной и чтобы работа нескольких приложений в параллельном режиме не приводила к порче данных. Это одна из задач, которую BerkeleyDB решает с помощью «окружений» (environment).

Если вы заглянете в директорию /var/jabberd/db/, то обнаружите там два файла authreg.db и sm.db, один или не-

сколько файлов типа log.0000xx, а также несколько файлов вида __db.00*. И что же с ними делать?

Директория /var/jabberd/db/ — это домашняя директория environment. Файлы с расширением *.db — это база данных (authreg — регистрационная информация, а sm — пользовательские данные). Файлы log.* — это, так сказать, оперативные поправки, внесённые в базу данных, которые по мере заполнения сбрасываются в файлы *.db. А файлы __db.* — это отображение областей памяти в дисковые файлы, которые и составляют environment, поскольку реализуют совместную работу нескольких приложений над базой данных. Тут следует заметить, что Jabberd2 состоит из нескольких независимых процессов, так что для него разделение доступа к базе данных весьма актуально.

Вся информация хранится в файлах *.db и в одном из log-файлов, в том, что активен на текущий момент. Длина log-файла фиксирована, по умолчанию она задаётся во время компиляции и составляет примерно 10 Мб, и может варьироваться с помощью файла настроек DB_CONFIG (о нём ещё пойдёт речь дальше).

По мере заполнения log-файла создаётся новый, с порядковым номером на единицу больше. Поскольку информация из старых log-файлов уже сброшена в файлы *.db, то они, вообще говоря, не нужны, так что в этом случае необходимо будет позаботиться о ротации логов, что также может осуществляться автоматически приложением, если только его автор об этом позаботился. Как вы понимаете, Jabberd2 — это не тот случай.

Все файлы — бинарные, так что их просмотр в текстовом редакторе или командой less вам мало что даст. Для этого случая предусмотрены специальные утилиты, например, db_printlog, которая позволит посмотреть последние правки, внесённые в базу данных (или по крайней мере время, когда они сделаны, поскольку остальная информация не слишком вразумительна).

Какие именно из log-файлов активны в данный момент, можно узнать с помощью команды:

```
# db_archive -l
log.0000000001
```


и удалить лишние с помощью:

```
# db_archive -d
```

Также эта утилита позволяет выяснить, какие файлы данных содержатся в каталоге (если вы забыли об этом):

```
# db_archive -s
```

```
authreg.db
sm.db
```

Вывод этих двух команд указывает, какие файлы нужны в архивировании при создании резервных копий — всего три штуки: log.0000000001, authreg.db, sm.db. Файлы вида __db.* системно зависимы и поэтому непереносимы, однако они могут быть восстановлены при наличии правильно созданной резервной копии.

Проще всего при архивировании не копировать файлы вручную, а использовать готовую команду:

```
# db_hotbackup -v -c -h ./ -b /root/backup
```

```
db_hotbackup: hot backup started at Sun May 4 00:29:55 2008
db_hotbackup: ./: force checkpoint
db_hotbackup: ./: remove unnecessary log files
db_hotbackup: copying ./authreg.db to /root/backup/authreg.db
db_hotbackup: copying ./sm.db to /root/backup/sm.db
db_hotbackup: copying ./log.0000000001 to /root/backup/log.0000000001
db_hotbackup: lowest numbered log file copied: 1
db_hotbackup: /root/backup: run catastrophic recovery
db_hotbackup: /root/backup: remove unnecessary log files
db_hotbackup: hot backup completed at Sun May 4 00:29:57 2008
```

Ключ -h указывает домашнюю директорию environment, ключ -b — директорию, куда осуществляется копирование. С помощью ключа -c перед копированием происходит сбрасывание последних изменений из log-файла в базы данных и удаление устаревших log-файлов. Кстати, принудительное сбрасывание данных можно осуществить с помощью команды:

```
# db_checkpoint -v -1
```

```
db_checkpoint: checkpoint begin: Sun May 4 00:32:31 2008
db_checkpoint: checkpoint complete: Sun May 4 00:32:31 2008
```

Только учтите, что на самом деле db_checkpoint — это не одноразовая операция, а демон, который, будучи запущенным без ключа -1, периодически проверяет состояние log-ов в текущей директории и по мере их заполнения или в определённые промежутки времени производит синхронизацию баз данных.

Содержимое директории с резервной копией сейчас выглядит таким образом:

```
# ls /root/backup
```

```
authreg.db log.0000000001 sm.db
```

В таком виде база данных не работоспособна. То есть после копирования необходимо будет её восстановить, предварительно переместив в ту же директорию конфигурационный файл DB_CONFIG (если вас не устраивают параметры по умолчанию) и запустив там команду:

```
# db_recover -c -v -h ./ -e
```

```
Finding last valid log LSN: file: 1 offset 522761
Recovery starting from [1][28]
Recovery complete at Sun May 4 00:49:52 2008
Maximum transaction ID 800003eb Recovery checkpoint [1][522761]
```

Здесь ключи запуска, помимо очевидного -v (verbose), означают, что команда делает восстановление после катастрофического завершения (-c) и восстанавливает окружение -e в текущей директории (-h ./). Последняя строка результата указывает, какой записи в log-файле соответствует точка синхронизации. С помощью db_printlog можно убедиться, что это предпоследняя запись в log-файле.

Теперь резервный каталог выглядит таким образом:

```
# ls ./
```

```
DB_CONFIG _db.002 _db.004 _db.006 log.0000000001
_db.001 _db.003 _db.005 authreg.db sm.db
```

Проверим результат:

```
# db_verify -h ./ authreg.db sm.db
```

Отсутствие сообщений об ошибках указывает, что копирование и восстановление базы данных прошло успешно.

По мере обновления программного обеспечения может возникнуть ситуация, что формат базы данных не соответствует новой версии библиотеки BerkeleyDB, которая была установлена на компьютере в процессе обновления. В этом случае нужно также обновить базы данных с помощью утилиты db_upgrade, соответствующей новой версии библиотеки:

```
# db_upgrade -v -h ./ authreg.db sm.db
```

```
db_upgrade: authreg.db upgraded successfully
db_upgrade: sm.db upgraded successfully
```

Последний вопрос, который обычно возникает у администраторов баз данных, — это оптимизация работы базы данных. Для BerkeleyDB решение этого вопроса сводится к написанию файла DB_CONFIG с параметрами environment, более отвечающими мощности и возможностям вашей системы.

С помощью перечисленных параметров можно управлять величиной дискового кэша и кэша в памяти, методом ведения логов и их размерами, механизмами блокировок и т. д.:

- set_cachesize
- set_data_dir
- set_flags DB_AUTO_COMMITIDB_CDB_ALLDIBIDB_DIRECT_DIBIDB_DIRECT_LOGIDB_DSYNC_DIBIDB_DSYNC_LOGIDB_LOG_AUTOREMOVEIDB_LOG_INMEMORYIDB_NOLOCKINGIDB_MULTIVERSIONIDB_NOMMAPIIDB_NOPANICIDB_OVERWRITEIDB_PANIC_ENVIRONMENTIDB_REGION_INITIDB_TIME_NOTGRANTEDIDB_TXN_NOSYNCIDB_TXN_SNAPSHOTIDB_TXN_WRITE_NOSYNCIDB_YIELDCPU
- set_lg_bsize
- set_lg_dir

- set_lg_max
- set_lg_mode
- set_lg_regionmax
- set_lk_detect DB_LOCK_DEFAULTIDB_LOCK_EXPIREIDB_LOCK_MAXWRITEIDB_LOCK_MINLOCKSIDB_LOCK_MINWRITEIDB_LOCK_OLDESTIDB_LOCK_RANDOMIDB_LOCK_YOUNGEST
- set_lk_max_lockers
- set_lk_max_locks
- set_lk_max_objects
- set_mp_mmapsize
- set_shm_key
- set_thread_count
- set_timeout
- set_tmp_dir
- set_tx_max
- set_verbose
- mutex_set_align
- mutex_set_max
- set_tas_spins
- rep_set_config DB_REP_CONF_BULKIDB_REP_CONF_DELAYCLIENTIDB_REP_CONF_NOAUTOINITIDB_REP_CONF_NOWAIT

Более подробную информацию по каждому из параметров можно получить в документации по соответствующей C функции из BerkeleyDB API в каталоге docs/api_c, где файлы руководства для этого класса функций имеют приставку env_. Так что, скажем, справка по set_cachesize находится в файле env_set_cachesize.html. Большинство параметров принимают в качестве значений одно или несколько строчных или целочисленных значений, что можно узнать в документации. Для случаев, когда выбор ограничен набором элементов, варианты допустимых значений приведены ранее. Например, можно использовать такой DB_CONFIG-файл:

```
# one 0GB 2MB cache
set_cachesize 0 2097152 1

# Transaction Log settings
set_lg_max 2097152
set_lg_bsize 262144
set_flags DB_LOG_AUTOREMOVE
set_flags DB_DIRECT_LOG
set_flags DB_DIRECT_DB
```

который увеличивает размер кэша до 2 Мб (стандартных 256 Кб для современных программ обычно всегда оказывается мало, что приводит к час-

тым обращениям к файлу базы данных на диске), ограничивает размер дискового log-файла 2 Мб, а log-файла в оперативной памяти – 262 Кб.

Кроме того, с помощью флагов конфигурируется автоматическое удаление устаревших логов, а также отключается системная буферизация доступа к файлам баз данных и логов (поскольку они и так буферизируются средствами BerkeleyDB).

Проверить эффект от изменения параметров по умолчанию можно с помощью команды «db_stats -e», которая позволяет получить суммарную информацию о размерах кэша, размерах log-файлов, количества выполненных и незаконченных транзакций, установленных локах и репликациях (по отдельности эту информацию можно получить с помощью ключей -m, -l, -t, -s и -r соответственно). Также эта команда с помощью ключей -s и -d выдаёт минимальные сведения о состоянии баз данных, например, количества записей в них.

Для полноты стоит упомянуть об уже встречавшейся в предыдущей части команде db_load. Хотя в предыдущей статье с помощью этой команды нам удалось создать и загрузить список пользователей authreg.db, однако для более сложных случаев создание текстового файла в формате, пригодном для загрузки в BerkeleyDB, оказывается ничем не легче, чем прямое редактирование баз данных с помощью программного интерфейса. В следующем разделе мы как раз и перейдём к программированию с помощью библиотеки BerkeleyDB.

В недрах sm.db

Иногда бывает нужно оперативно изменить данные в базах данных BerkeleyDB или посмотреть уже имеющуюся там информацию. Если для решения первой части задачи можно прибегнуть к команде db_load, то со второй частью не всё так просто. К сожалению универсального инструмента для этой цели нет и быть не может, что связано именно с присущими BerkeleyDB ограничениями. Но эта задача может быть решена с помощью написания скрипта для каждого конкретного случая, в данном случае нашей целью будет создание «официального утверждённого списка контактов»

(published roster), хранящегося на стороне сервера. Для этого нам придётся вручную отредактировать базу данных sm.db, которая содержит динамически изменяемые настройки Jabberd2 и используется, в основном, компонентом sm. В отличие от authreg.db её не нужно создавать до запуска сервера; создаётся она при регистрации первого пользователя, а затем пополняется по большей части в автоматическом режиме.

Попутно нам придётся немного глубже разобраться с устройством BerkeleyDB. В качестве рабочего инструмента будем использовать скриптовый язык TCL, который завоевал популярность благодаря простоте своего синтаксиса («все есть строка, а каждая инструкция начинается с имени процедуры, за которым следует список фактических параметров»).

Более того, BerkeleyDB имеет встроенный интерфейс под этот язык, наравне с C++ и Java, сборка которого активируется выбором параметра --enable-tcl при компиляции библиотеки. Более полное представление об архитектуре и программировании BerkeleyDB можно получить из книги [2], однако там в качестве рабочего языка при разборе примеров взят C++.

В качестве напоминания в одном абзаце будут перечислены основные элементы языка Tcl. Этого хватит, чтобы разобраться с приведёнными ниже скриптами.

Tcl – это фактически shell, у переменных типов нет, всё является строкой. Каждая команда Tcl представляет собой вызов функции, за которым следует список параметров. Например, создание собственной функции:

```
proc myfunction { z } {
    puts [set z]
}
```

представляет собой вызов функции proc, которой передано три параметра – имя определяемой функции myfunction, список формальных параметров, состоящий из одного элемента z и тела процедуры, в котором выдаётся на печать значение параметра. Фигурные скобки являются группирующими элементами, квадратные означают выполнение функции и подстановку результата. То есть в дан-

ном случае [set z] приводит к подстановке вместо переменной её значение. В качестве синтаксической глазури в tcl для той же самой цели используется более привычный \$. Команда set используется также для присваивания set z 1, например.

BerkeleyDB устроена более примитивно, чем более популярные иерархические или реляционные базы данных, поскольку все данные в BerkeleyDB хранятся в виде списков пар – «ключ-значение». Если проводить аналогию с реляционными базами данных, то мы имеем как бы таблицу с одной колонкой – первичным индексом, и второй – значением. В то же время именно простота устройства BerkeleyDB позволяет использовать её в качестве строительного блока для создания как иерархических (OpenLDAP), так и реляционных (MySQL) баз данных.

Вторым отличительным моментом BerkeleyDB является отсутствие типизации данных, так что в качестве ключей и их значений могут быть использованы любые типы данных, в том числе массивы, списки, если только приложения, подключённые к базе данных, умеют их правильно интерпретировать. С одной стороны, это даёт большой простор для разработчиков приложений, так как они не ограничены в выборе структур данных. А с другой – усложняет обмен данными между приложениями, поскольку для работы с базой данных приложение должно изначально точно знать используемые типы структур. Собственно поэтому и не существует универсальных утилит для доступа к базам данных BerkeleyDB. Вообще-то, и использо-

вание db_load применимо только для баз данных, содержащих только строковые значения, так что нам в некотором смысле повезло, что authreg.db не содержит записей другого типа.

Тип доступа и механизм индексации в базах данных можно выбрать при создании базы данных. Всего BerkeleyDB поддерживает 4 типа – Queue, Recno, Hash, BTree. Условно говоря, эти типы означают, каким именно образом BerkeleyDB индексирует и хранит базы данных. Все базы, используемые в Jabber2, представляют собой хэши.

Любой файл базы данных BerkeleyDB (*.db) на самом деле может содержать несколько «таблиц»/списков. Чтобы посмотреть, что представляет собой база данных sm.db изнутри, откроем её с помощью скрипта list-sm.tcl:

```
load /usr/lib/libdb tcl-4.4.so
[berkdb env -home ./ -recover] close
set e [berkdb env -data dir ./ -home ./]
set h [berkdb open -rdonly -env $e sm.db]
set c [$h cursor]
catch {
  for {set entry [$c get -first]} { $entry != {} } {
    { set entry [$c get -next]} {
      eval entry $entry
      puts [lindex $entry 0]
    }
  }
}
$c close
$h close
$e close
```

Идея проста. Вначале подгружается интерфейс доступа к библиотеке BerkeleyDB, а потом открывается environment,

«АКАДЕМИЯ КОРПОРАТИВНЫХ СИСТЕМ» Центр обучения и сертификации ИТ-специалистов

Лучшие условия обучения для системных администраторов

Дневная и вечерняя форма обучения
Тестирование по будням и в выходные дни

Подготовка к международной сертификации:
Microsoft, Cisco, Linux и др. Тестирование

Более 400 курсов по направлениям:

- ✓ **Системное администрирование**
Microsoft, Linux, FreeBSD, Sun Solaris
- ✓ **Администрирование и разработка баз данных**
на Microsoft SQL Server и Oracle
- ✓ **Работа с сетевым оборудованием Cisco**
- ✓ **Разработка и поддержка приложений** в среде Microsoft Visual Studio .Net
- ✓ **Технологии защиты информации в сетях** под управлением Microsoft, Linux, FreeBSD
- ✓ **Управление проектами** по методологиям PMI, IPMA с использованием Microsoft Project



В честь юбилея учебного центра для системных администраторов действует акция

«5 лет успеха – подарки всем!»

со специальными условиями и ценами

Реклама

www.a-sys.ru

(495) 748-37-41

Мы находимся в центре Москвы: 5 минут пешком от м. «Пролетарская»

Рисунок 1. Электронная визитка

что (как вы помните) позволяет безопасным образом контролировать доступ к базам данных.

На самом деле окружение открывается дважды, первый раз в режиме восстановления, что является рекомендуемым способом обработки ошибок, вызванных возможным некорректным завершением работы при предыдущем открытии базы данных. Затем в уже созданном окружении открывается интересующая нас база данных, причём делается это в режиме read-only, поскольку для баз, содержащих несколько таблиц, иного выбора нет (позже вы увидите, как открыть доступ к отдельным таблицам на запись).

Далее в базе данных создаётся курсор-итератор, и в цикле последовательно перебираются все ключи, которые представляют собой имена списков. Вся логика работы с базой данных собрана в операторе перехвата ошибок catch, что гарантирует корректный переход к закрытию всех открытых дескрипторов точек доступа к файлам и базам данных при возникновении ошибок.

Запускается скрипт в директории, где находится файл sm.db, таким образом:

```
$ tclsh ../scripts/list-sm.tcl
active
disco-items
logout
motd-message
privacy-default
privacy-items
queue
roster-groups
roster-items
status
vacation-settings
vcard
```

Так выглядит список «таблиц» в только что созданной базе данных. Здесь и далее предполагается, что все скрипты собраны в отдельной директории /var/jabberd/scripts. Обратите также, что в скрипте нужно указывать ту версию библиотеки BerkeleyDB, с которой фактически собран Jabberd2. Как правило, в системе (для совместимости) присутствуют несколько версий этой библиотеки, так что тут нужно быть внимательным.

Попробуем теперь продвинуться дальше и прочитать из таблицы vcard список виртуальных визитных карточек с информацией о пользователях, которую они сочли нужным опубликовать в Jabberd. Впоследствии, например, можно будет сделать скрипт, который переносил бы эту информацию или её часть на LDAP-сервер.

Чтение таблицы осуществляется скриптом list-vcard.tcl:

```
source ../scripts/serialize.tcl
load /usr/lib/libdb tcl-4.4.so
[berkdb env -home .7 -recover] close
set e [berkdb env -home ./]
set h [berkdb open -env $e sm.db vcard]
set c [$h cursor]
catch {
    for {set entry [$c get -first]} { $entry != {} } {
        { set entry [$c get -next]} {
            eval set vcards $entry
            puts "->key(jid) = [lindex $vcards 0]"
            set card [deserialize [lindex $vcards 1]]
            foreach {field value} $card {
                puts "$field : $value"
            }
        }
    }
}
$c close
$h close
$e close
```

Как видите, по сравнению с предыдущим скриптом изменений немного. Во-первых, на этот раз открывается только одна из «таблиц» (vcard) базы данных sm.db, и, во-вторых, добавлена специальная обработка значений с помощью дополнительного файла serialize.tcl. Хотя как ключ, так и данные в BerkeleyDB могут быть любого типа, но разработчики Jabberd2, как правило, используют в качестве ключей строки, а для значений используют список-хэш. Формат списка выглядит так – вначале идёт имя элемента списка в виде строки, заканчивающейся нулём, затем идёт целое число, которое обозначает тип элемента, затем его значение, и так для каждого из элементов.

Функция deserialize как раз и служит для разбора записей, выполненных в таком формате, и возвращает их в виде парного Tcl-списка, который потом группируется парно через разделитель-двоеточие и выводится на экран. Вот, например, моя электронная визитка (та же самая, что и на рис. 1):

```
$ tclsh ../scripts/list-vcard.tcl
->key(jid) = mike@hppt.troitsk.ru
nickname : mike
email : mkondrin@hppt.troitsk.ru
fn : Kondrin Mikhail
adr-region : MO
adr-country : RU
n-given : Mikhail
n-family : Kondrin
```

До настоящего Idif-формата, пригодного для загрузки на сервер LDAP, пока ещё далеко, но направление движения, по крайней мере, понятно.

Не стоит думать, что BerkeleyDB поддерживает только простые таблицы. Поскольку BerkeleyDB позволяет осуществлять объединение (join) по первичному и вторичному индексам, в нем можно хранить данные достаточно сложной структуры. Также замечательной чертой BerkeleyDB является наличие транзакций, что позволяет вносить изменения в несколько таблиц атомарным способом, в стиле «все или ничего».

Как раз для решения поставленной вначале задачи – заполнение `published-roster` – нам и придётся столкнуться с транзакциями. В данном случае это будет скорее вынужденной мерой, поскольку в `Jabberd2` доступ ко всем таблицам из `sm.db` возможен только в режиме транзакций. Чтобы `Jabberd2` мог использовать созданную нами таблицу, в точно таком же режиме нам и следует открыть `published-roster`. Все элементы в таблице имеют ключ, состоящий из пустой строки, а в качестве значений используется хэш-список столбцами `ask from to` (целочисленными) `jid name group` (строкового типа). Эта информация построчно заносится в текстовый файл `published.txt`, откуда она и будет впоследствии считываться скриптом. Например, в нашем случае можно предложить такой вариант этого файла:

```
root@myrealm.ru 0 0 0 Admin root
postmaster@myrealm.ru 0 0 0 Admin postmaster
mike@myrealm.ru 0 0 0 Colleagues mike
```

Все учётные записи разнесены по двум группам – реальным пользователям (группа Colleagues) и административно-техническим аккаунтам (Admin). Скрипт выглядит таким образом:

```
source ./scripts/serialize.tcl
load /usr/lib/libdb tcl-4.5.50
[berkdb env -home .7 -recover] close
set e [berkdb env -home ./]
set b [berkdb open -dup -hash -create -auto_commit -env $e
        $e sm.db published-roster]
set f [open published.txt]
set t [$e txn]
set err { catch {
    foreach line [split [read $f] "\n"] {
        set a [ pub_roster -format $k]
        $b put -txn $t "" $a
    }
}}
if { $err } {
    $t abort
} else {
    $t commit
}
close $f
$b sync
$b close
$e close
```

Открытие базы данных sm.db происходит в транзакционном режиме за счёт выставленного флага `auto_commit`, она имеет тип `Hash` и не требует уникальности индекса (`-dupl`). Все изменения, вносимые в файл, обернуты в транзакцию, причём в отличие от предыдущих случаев вызов `catch` анализируется на предмет обнаружения ошибок, и (в зависимости от результата) транзакция либо откатывается, либо нормальным образом завершается после прочтения файла и заполнения `published-roster`. Затем база данных синхронизируется, т.е. все изменения, хранящиеся в кэше, в памяти, принудительно сбрасываются на диск. Сама процедура форматирования записей `pub roster format` для по-

мещения в базу данных выглядит довольно уродливо, и поэтому вынесена также в подгружаемый файл `serialize.tcl`. В итоге результат выглядит примерно так, как показано на **рис. 2**.

На этом зада-
чу можно считать
решённой. При вхо-
де на сервер каждый
пользователь теперь
должен видеть от-
дельные общие груп-
пы со списком «опубл-
администратор может
ведённого выше. Сле-
ние именно локально
но с «техническим зад-
тавили. Если же, скаж-
LDAP с электронными
то вам, скорее, имеет
хранилища v-cards. Ja-
ношении и позволяет
для каждого из моду-
фигурационном файле
показано, каким обра-
ток и published roster в
ных данных, для кото-

И в завершение небольшая ложка дёгтя. Как уже говорилось, основной недостаток Jabberd2 — это практически полное отсутствие документации и несоответствие уже имеющихся руководств с реальным положением дел. В эту же ловушку попал и я сам. Этот цикл статей был написан в расчёте на серию релизов Jabberd2 2.1.x, последний из которых (2.1.24.1) вышел в апреле этого года. Однако уже в следующем релизе (2.2.0) произошли изменения в архитектуре Jabberd2, в результате чего схема, приведенная в предыдущей статье, уже не соответствует действительности. Компонент `resolver` был упразднен, а вся его функциональность перенесена в компонент `s2s`. При этом единственный информативный блок `<lookup>` из его конфигурационного файла также перемещен в файл `s2s.xml`. В общем-то, для динамично меняющегося проекта такие резкие изменения не являются чем-то необычным, но хотелось бы, чтобы эти изменения хоть каким-то образом находили свое отражение в документации, хотя бы на уровне файлов `README` (где по-прежнему упоминается `resolver`, как один из независимых компонентов) и `ChangeLog`. К сожалению, этот призыв можно адресовать не только авторам Jabberd2, но и многим другим разработчикам свободного программного обеспечения. 🍌

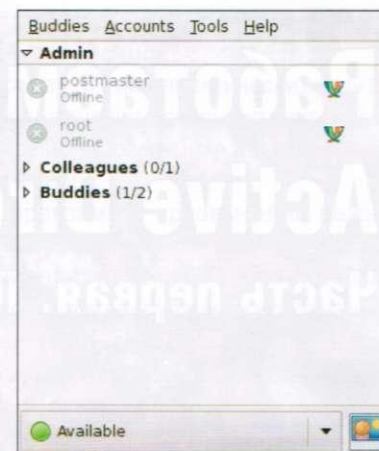


Рисунок 2. Открытая база данных

1. Кондрин М. Jabberd2 – простой и нетребовательный к ресурсам XMPP-сервер. //Системный администратор, №8, 2008 г. – С. 34-40.
2. Himanshu Yadava. The Berkeley DB Book. Apress, 2007.

Работаем с данными Active Directory из скриптов

Часть первая. Применение Bourne Shell

Рашид Ачилов

В корпоративной среде при разработке скриптов рано или поздно возникает вопрос: «А неплохо было бы иметь базу всех сотрудников с именами, должностями, адресами и номерами ICQ...», что зачастую приводит к самостоятельным разработкам разной степени удачливости. Но есть и более простой способ. Такая база уже есть. Ее за нас придумала Microsoft. Это Active Directory. Нужно только уметь с ней общаться.

Просто LDAP, и ничего особенного

В статье не будет рассматриваться, что такое Active Directory и как она работает – на эту тему написано множество книг, например [1, 2], а также описание принципов работы Active Directory (договоримся в дальнейшем

называть ее AD) в серьезных справочниках для администраторов Windows, например [3]. Мы обсудим здесь только те особенности, которые пригодятся нам в дальнейшем.

Для тестирования всюду будет использоваться Windows 2003 Server с именем AD shelton.net, коротким име-

нем (pre-Windows 2000) «CATS», именем самого компьютера BIGCAT.

Человек, никогда не видевший AD получив сетевой доступ к серверу AD непременно воскликнет: «Так ведь это же сервер LDAP!». И будет прав. Действительно, со стороны сети сервер AD выглядит как сервер LDAP.

Вот какие открытые порты можно обнаружить на нем:

```
Starting Nmap 4.62 ( http://nmap.org ) at 2008-08-01 01:45 NOVST
Interesting ports on bigcat (192.168.50.1):
Not shown: 1699 closed ports
PORT      STATE SERVICE
40/tcp    open  nameserver
53/tcp    open  domain
88/tcp    open  kerberos-sec
135/tcp   open  msrpc
136/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  microsoft-ds
464/tcp   open  kpasswd5
593/tcp   open  http-rpc-epmap
636/tcp   open  ldaps1
636/tcp   open  LSA-or-nterm
8080/tcp   open  IIS
8080/tcp   open  inst1_boots
8080/tcp   open  globalcatLDAP
8080/tcp   open  globalcatLDAPssl
8080/tcp   open  ms-term-serv
```

Это типичный набор сервисов для контроллера домена (domain controller, dc). Он может слегка варьироваться, но сервис, который нас интересует – ldap (порт 389), – обязан присутствовать. Неизвестно, почему Microsoft решила не изобретать свой велосипед, как поступает обычно, а использовало хорошо известный и хорошо документированный LDAP. Зато над созданием схемы LDAP она «поработала» изрядно – скажем, на схему inetOrgPerson ее схема непохожа совершенно – типичный объект AD «пользователь» с точки зрения ldapsearch выглядит вот таким образом:

```
dn: CN=LDAP Reader,CN=Users,DC=shelton,DC=net
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: user
cn: LDAP Reader
cn: Reader
description: LDAP
displayname: CN=LDAP Reader,CN=Users,DC=shelton,DC=net
instanceType: 4
whenCreated: 20080817091820.0Z
whenChanged: 20080817091820.0Z
displayName: LDAP Reader
objectGUID: 954394
objectGUID: 954399
name: LDAP Reader
objectGUID:: S2W6nZo2XkCfS2jjHt4XBg==
userAccountControl: 66048
badPwdCount: 0
codePage: 0
countryCode: 0
lastLogonTime: 128634384464218750
lastLogoff: 0
lastLogon: 128634384682968750
passwordSet: 128634383003906250
primaryGroupID: 513
objectSid: AQuAAAAAAUVAAtInjRe4dmI0lvJgpVwQAAA==
accountExpires: 9223372036854775807
supplementalAttributes: {}
objectClass: ldapread
objectClass: 805306368
principalName: ldapread@shelton.net
objectCategory: CN=Person,CN=Schema,CN=Configuration,DC=shelton,DC=net
```

Это, разумеется, далеко не полный перечень полей объекта, а только те, что создаются при создании учетной записи пользователя с минимальным вводом данных о нем.

Ну и раз мы условились считать AD «просто LDAP», то, естественно, к ней будут применимы все средства тестирования для работы с LDAP, например, ldapsearch, программа из комплекта OpenLDAP Toolkit, а также ldapbrowser –

программа, написанная на Java (Swing), для работы с LDAP в графическом режиме (см. рисунок).

Что касается ldapbrowser, то с ним практически сразу все ясно (кроме настроек подключения, которые станут ясными сами по себе после описания ldapsearch). А вот о ldapsearch хотелось бы поговорить особо.

Ldapsearch – это программа, чрезвычайно важная для администратора LDAP, примерно такая же, как ping и traceroute для администратора сетевого. Она позволяет получить, а после несложных манипуляций и использовать практически любые данные, что есть в LDAP. Но...

У нее собственный язык запросов. Он не то что бы является сильно сложным, но повергает в изумление любого, кто сталкивается с ним в первый раз. Он построен в соответствии с принципом «обратной польской записи», как в старинных программируемых калькуляторах, когда сначала задается знак операции, а потом перечисляются операнды. Все операнды разделяются отдельными скобками, из-за этого выражения приобретают невероятную громоздкость. Поскольку это язык запросов, то в нем присутствуют только логические операции – И (&), ИЛИ (|), НЕ (!). Не путайте ИЛИ и НЕ, несмотря на то, что они сходны по начертанию! Например:

- (description=*) – любой объект, у которого определено поле description. При этом совершенно не важно будет, сколько и каких в поле символов – один символ «?» точно так же попадет под этот фильтр, как и «длинная-длинная строка».
- (&(objectClass=user)(mail=*)) – любой объект, у которого поле objectClass равно user и определено поле mail.
- (&(objectClass=user)(!(mobile=100*)(mobile=200*))) – любой объект класса user, у которого поле mobile начинается с цифр 100 или 200.
- (&(sAMAccountName=*)(homeMTA=*)(!(title=*)(department=*))))) – любой объект, у которого определены поля sAMAccountName и homeMTA и не определены поля title или department. Обратите внимание на расположение знаков логических операций и баланс скобок в последнем примере.
- (&(&(&(mailnickname=*)(!(objectCategory=person)(objectClass=user)(!(homeMDB=*))(!(msExchHomeServerName=*)))(!(objectCategory=person)(objectClass=user)(!(homeMDB=*)(msExchHomeServerName=*)))(!(objectCategory=person)(objectClass=contact))(objectCategory=group)(objectCategory=publicFolder)))))) – этот поистине чудовищный запрос используется в KAddressbook для получения глобальной адресной книги из LDAP. Вы можете попрактиковаться в его разборе самостоятельно. Создавал его не я, мне его посоветовали в рассылке, посвященной KDE PIM. Тот, кто его создавал, явно не знал ничего про структуру AD, потому что то же самое можно добиться запросом (&(sAMAccountName=*)(!(sAMAccountType=805306368)(sAMAccountType=268435456))(mail=*)). В приводимых числах в sAMAccountType нет ничего магического – это просто 0x30000000 и 0x10000000 в десятичной форме.

Более подробно язык запросов описан в [5]. С языком запросов немного разобрались. Остается сле-

дующий вопрос – как подключиться к LDAP? Здесь надо учесть несколько вещей.

Во-первых, в Windows Server отсутствует так называемый anonymous bind, то есть возможность подключения к LDAP без предъявления какого-либо имени и пароля. Это, конечно, можно включить, но по умолчанию это выключено. Поэтому перед тем, как начать работать с LDAP через ldapsearch или разрабатывать скрипты, необходимо создать отдельную учетную запись, отобразить у нее все права, какие только можно, установить достаточно простой пароль, не содержащий символов, которые при передаче через шелл могут быть интерпретированы (прямые и обратные слэши, амперсанд, знак вопроса и т. д.). Я обычно использую пароль «qwerty{123}» – почему-то системой Windows 2003 Server он считается достаточно сложным и имя пользователя ldapread. Для ldapsearch (и далее повсюду) в параметре, задающем так называемый bind dn, указывается строка <имя_пользователя>@<ДОМЕН>, причем с указанным соблюдением регистра, например, ldapread@SHELTON.NET. <ДОМЕН> – это имя AD, переведенное в верхний регистр.

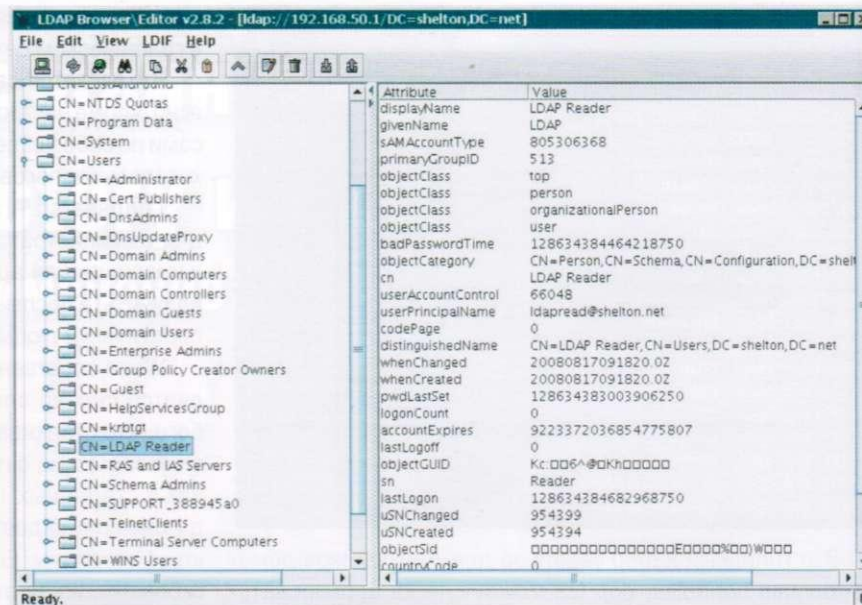
Во-вторых, что указывать в качестве корня для подключения (base dn)? Обычно base dn формируется следующим образом: если AD называется, допустим, shelton.net, то соответствующее значение base dn будет dc=shelton,dc=net. Если же оно называется, например, kitten.cats.shelton.net (такие длинные имена возможны в больших инсталляциях), то соответствующее значение base dn будет dc=kitten,dc=cats,dc=shelton,dc=net. Разумеется, можно при подключении указать не корень AD, а какую-либо его ветку, например, CN=Computers,DC=shelton,DC=net, но в таком случае поиск никогда не поднимется выше ее границы. Проблема же состоит в том, что логически правильным является создание организационных единиц (OU – Organizational Unit), допустим, соответствующих разным филиалам непосредственно в корне AD, поскольку эти ou будут содержать объекты разных типов – пользователей, компьютеры, группы, другие OU. Поэтому для просмотра всей AD лучше указывать корень.

В третьих, как в AD хранится информация, заносимая не ASCII символами, то есть на национальных (и на русском в том числе) языках? Хранится она в виде текста в кодировке UTF-8, преобразованного с помощью Base64 в набор символов ASCII. Подробности кодирования изложены в [6]. В том, что там именно Base64, мы вскоре убедимся.

Теперь пора попробовать ldapsearch в деле.

```
# ldapsearch -D ldapread@SHELTON.NET -w qwerty{123} -L
-L -LLL -h 192.168.50.1 -b dc=shelton,dc=net -P 3 -a
-a always (sAMAccountName=id*) distinguishedName sAMAccountType
```

```
dn: CN=LDAP Reader,CN=Users,DC=shelton,DC=net
sAMAccountType: 805306368
```



Ldapbrowser – программа работы с LDAP

Использованные ключи имеют следующее значение:

- -D – задает binddn;
- -w – пароль для подключения;
- -LLL – обеспечивает максимальное упрощение выводимой информации;
- -h – адрес сервера;
- -b задает basedn;
- -P – версия протокола LDAP;
- -a – описан в man ldapsearch.

Далее идет фильтр отбора записей (простейший) и перечень атрибутов, которые нужно вывести. Если не указать ничего, будут выведены все атрибуты. В данном случае мы запросили атрибуты distinguishedName и sAMAccountType. Обратите внимание, что атрибуты, которые необходимо вывести, перечисляются через пробел, а не через запятые.

Теперь добавляем в информацию о пользователе (средствами Microsoft) поле title, равное «Читатель», и запрашиваем его через ldapsearch:

```
# ldapsearch -D ldapread@SHELTON.NET -w qwerty{123} -L
-L -LLL -h 192.168.50.1 -b dc=shelton,dc=net -P 3 -a
-a always (sAMAccountName=id*) title sAMAccountType
```

```
dn: CN=LDAP Reader,CN=Users,DC=shelton,DC=net
title: 0KFQuNGC0LDRgtC10LvRjA==
sAMAccountType: 805306368
```

Здесь атрибут title закодирован согласно [6], что легко узнается по двум знакам «=». Проверяем:

```
# echo 0KFQuNGC0LDRgtC10LvRjA== | mmencode -u | iconv -L
-f utf-8 -t koi8-r
```

```
Читатель
```

Как мы видим, [6] не обманывает нас – это действительно строка UTF-8, закодированная через Base64. Этот прием – перекодирование через mmencode и преобразование через iconv – это уже практически готовая часть скрипта, к рассмотрению которого мы сейчас и приступим.

Работаем из Bourne Shell

Разумеется, решать «задачу ради задачи» скучно. Поэтому возьмем вполне конкретный пример – для использования программой Sarg сформировать так называемый usertab-файл, который задает соответствие логинов, использованных в процессе аутентификации в Squid и их имен, так как они будут выведены в отчете, так что получается вроде:

```
ivanov-iv      Иванов Иван
petrov-pp      Петров Петр
root           Администратор
```

Последовательность действий будет практически такая же, как мы только что проводили вручную – подключаемся к LDAP с соответствующим фильтром и списком полей, получаем вывод, преобразовываем его и сохраняем в виде файла. Нас интересуют только поля displayName и sAMAccountName. Поскольку это демонстрационный скрипт, а не полная версия (полная версия содержит больше элементов и выполняется в соответствующем окружении), то некоторые части его будут опущены – они в принципе уже не раз приводились и могут быть взяты из любого другого скрипта с [4]. Рассматриваться будут только те части, которые непосредственно отвечают за выполнение интересующих нас действий. Так, например, не будут рассматриваться функции usage() и safe_logger().

Данный скрипт будет использовать конфигурационный файл типового формата, в котором собраны настройки подключения к LDAP:

```
ldap_server=192.168.50.1
ldap_basedn="dc=shelton,dc=net"
ldap_binddn=ldapread@SHELTON.NET
ldap_password="cXdlcnR5YXNkZgo 1"
ldap_common_filter="(sAMAccountName=*) & (sAMAccountType=805306368) (telephoneNumber=*)"
etcdir=/usr/local/etc
sargdir=sarg2
sargfile=sargusers
```

Здесь все должно быть понятно. Кроме пароля. Для пароля применена такая странная запись по следующим причинам. Во-первых, то, что хранить пароли в открытом виде – это, знают все. Но в то же время трудно найти обратимую функцию для того, чтобы надежно зашифровать пароль, и потом столько же надежно его расшифровать, не прибегая к помощи PGP. Поэтому был выбран компромисс – простейшее преобразование формата, защищающее от случайного взгляда, и не более. Это именно преобразование, а не шифр. Точнее говоря, это Base64, у которого отброшены заполнители и вместо них проставлено их число, чтобы слегка запутать. Разворачивается этот пароль вот такой несложной процедурой:

```
# Обратное преобразование строки пароля
# Вход: $1 enconvd (строка) - пароль из файла конфигурации
# Выход: _passwd - пароль в текстовом виде
demux_passwd()
{
    local ideconv _iadd _ifill
    _ifill=""
    _ideconv=$1
    if [ ! $2 -eq 0 ]; then
        _iadd=$2
    while [ $_iadd -gt 0 ]; do
        _ideconv=$_ideconv$_ifill
        _iadd=$(($_iadd-1))
    done
```

```
done
fi
_passwd=`echo $_ideconv | $util_mmencode -u`
}
```

Пояснять тут особо нечего – если второй параметр не ноль, то циклом дописать в конец строки столько знаков заполнения, сколько передано во втором параметре, и декодировать строку.

Основной принцип работы скрипта следующий: после вводной части (загрузки конфигурационного файла, разбора командной строки, поиска необходимых программ) скрипт «расшифровывает» пароль, создает временный каталог для работы, потом запрашивает из LDAP логины и имена пользователей и сохраняет их во временных файлах (процедуры описаны ниже и довольно подробно):

```
safe_logger "LDAPQuery ver. $revisionNumber started"
# Обратное преобразование пароля
demux_passwd $ldap_password
ldap_pwd=$_passwd
# Создать временный каталог, если еще не создан,
# и перейти в него
make tempdir
# Выполнить поиск в LDAP и сохранить результат
ldap_data_query sAMAccountName $tempdir/users.ldif
# Преобразовать данные в читаемый вид
ldap_data_deconvert $tempdir/users.ldif $tempdir/users.list
# Выполнить поиск в LDAP и сохранить результат
ldap_data_query displayName $tempdir/names.ldif
# Преобразовать данные в читаемый вид
ldap_data_deconvert $tempdir/names.ldif $tempdir/names.list
```

Далее выполняется прием, который я применяю очень часто – два вложенных цикла сравнения двух списков. Первый список – это список полученных из LDAP логинов, второй – список логинов пользователей, уже присутствующих в файле. Он получается тривиальной командой на awk. Суть его в том, что оба списка считываются в переменные, берется один элемент «внешнего» списка и сравнивается со всеми по очереди элементами «внутреннего» списка. Когда находится совпадение, выполняются необходимые действия, потом внешний цикл переходит к следующему элементу (в Bourne Shell такое возможно – указать оператору break или continue число уровней вложенности, на которые он одновременно воздействует).

Во внешнем цикле, идущем по списку логинов из LDAP, обнуляется счетчик позиции, который будет отмечать позицию имени в списке имен. Внутренний же цикл идет по списку пользователей, уже присутствующих в файле. Если логин из LDAP уже присутствует в файле, то незачем его искать дальше, и переходим к следующему логину. Если же отсутствует, то берется имя пользователя из списка имен, отступив с помощью tail на значение счетчика позиции и взяв оттуда одну строку, после чего формируется строка вывода и выводится в список пользователей Sarg с дозаписью:

```
# Внешний цикл по списку пользователей из LDAP
for _onedapuser in $ldapusers
do
    togo=0
    # Внутренний цикл по списку пользователей Sarg
    for _onesarguser in $sargusers
    do
        # Если пользователь найден, то мы прерываем
        # и внутренний, и внешний циклы.
        # Переходим к следующей записи внешнего цикла
        if [ $_onedapuser = $_onesarguser ]; then
            continue 2
        fi
    done
done
```



```

fi
done
# Если мы находимся здесь, значит, пользователь LDAP
# не найден в списке пользователей Sarg. Мы записываем
# в список пользователей Sarg строку с логином
# из names.list и именем из names.list
_oneusername='cat $tempdir/names.list | \
tail -n +$togo | head -n 1'
echo -e "$_onedapuser \t$_oneusername\n" >> \
    $setcdir/$sargdir/$sarglist
togo=$(( $togo+1 ))
added=$(( $_added+1 ))
done

```

Наибольший интерес, конечно, представляют процедуры `ldap_data_query()` и `ldap_data_deconvert()`.

`ldap_data_query()` – достаточно простая процедура, которая всего лишь вызывает `ldapsearch`, передавая ему все необходимые параметры из конфигурационного файла и списка переданных параметров. Единственное, что здесь делается до вызова `ldapsearch`, – это замена запятых в списке запрашиваемых полей на пробелы:

```

# Запросить некоторые данные у LDAP-сервера
# Вход: $1 fieldname (строка) - Имя атрибута(ов),
# запрашиваемых в AD
# $2 outfile (строка) - Имя файла для сохранения
# полученных данных
# Выход: none
ldap_data_query()
{
    local ldapfield outfile
    _ldapfield=echo $1 | sed -e "s::: :g"
    outfile=$2
    # Выполнить запрос к LDAP и сохранить результат
    $util_ldapsearch -T $tempdir -D $ldap_binddn \
        -w $ldap_pwd -LLL -h $ldap_server \
        -b $ldap_basedn -P 3 -a always \
        $ldap_common_filter $_ldapfield > $_outfile
    # Проверить код возврата
    status=$?
    if [ $status -ne 0 ]; then
        safe_logger "Unable to complete LDAP request to get \
            $_ldapfield from AD"
        exit
    fi
}

```

А вот процедура `ldap_data_deconvert()` гораздо интереснее. Предварительно рассмотрим вспомогательные процедуры `unspace()` и `make_tempdir()`.

```

# Удалить пробелы и строки комментариев из файла
# Вход: $1 source (строка) - Имя исходного файла
# $2 destination (строка) - Имя файла результата
# Выход: none
unspace()
{
    cat $1 | grep -v ^# | sed -e "s::: :g" > $2
}

```

Процедура `unspace()` удаляет из файла, заданного первым параметром, все строки, начинающиеся со знака «#», а также удаляет все пробелы. В итоге на выходе будут только строки вывода `ldapsearch`:

```

# Создать временный каталог, если он еще не был создан,
# и перейти в него
# Вход: none
# Выход: tempdir (строка) - имя временного каталога
make_tempdir()
{
    if [ $#tempdir -eq 0 ]; then
        tempdir='mktemp -d /tmp/adphones.XXXXXX'
    fi
    cd $tempdir
}

```

`make_tempdir()` исключительно проста – она создает временный каталог с уникальным именем, если такового еще не было создано. Сделано это для удобства использования данного скрипта из других скриптов, чтобы избежать гонок (race condition).

Наиболее сложными для понимания являются две основные процедуры – `ldap_data_deconvert()` и `write_when_not_dn()`. В чем их смысл?

Дело в том, что `ldapsearch` в обязательном порядке выводит строку `dn` для того объекта, поля которого печатаются. Эту строку необходимо пропустить. Но здесь мы имеем еще одну проблему: если в `dn` есть символы национальных алфавитов, то оно будет представлено `base64`-кодом. Поскольку `dn` может быть очень длинным, то кодировка его зачастую значительно превышает по длине 80 символов, а у `ldapsearch` есть еще одна «особенность» – после 80 позиций текста обязательно ставится символ перевода строки, так что просто разобрать по строкам не получается. И выбрасывать переводы строк нельзя – тогда пропадет разбиение между атрибутами. Поэтому и пришлось сделать несколько нетривиальных ходов. Для описания `ldap_data_deconvert()` скажем, что `write_when_not_dn()` выводит данные, требующие преобразования в один файл, не требующие – в другой и пропускает поле `dn` при чтении. Кроме того, есть еще одна очень важная для нас особенность `ldapsearch` – если атрибут содержит данные в `base64`, то значение отделяется от имени атрибута двумя символами двоеточия вместо одного, то есть если есть второе поле – это данные, которые нет необходимости перекодировать, если его нет, но есть третье – это данные, которые необходимо перекодировать.

Собственно же `ldap_data_deconvert()` выглядит следующим образом:

```

# Обратное преобразование данных, прочитанных из AD
# Вход: $1 source (строка) - имя файла с исходными
# данными
# $2 dest (строка) - имя файла для записи
# выходных данных
# Выход: none
ldap_data_deconvert()
{
    local tempsfx outfile tempstr enconvded skip
    # Из строки типа /path/to/filename.ext мы выбираем
    # только имя файла. Первая команда отбрасывает путь
    # (шаблон - наибольший префикс). Вторая команда
    # отбрасывает расширение (шаблон - наименьший суффикс)
    tempsfx=${1##*/}
    _tempsfx=${_tempsfx%.*}
    outfile=$2
    # Удаляем из файла все пробелы и строки комментариев
    unspace $1 stripped.$_tempsfx
    # Готовимся разбивать по строкам, для чего используем
    # глобал $newline, равный переводу строки
    saveifs=$IFS
    IFS=$newline
    _skip=0
    _tempstr=""
    _enconvded=0
    _convded='cat stripped.$_tempsfx'
    for one in $convded
    do
        # Внутри строки разбиваем по символу двоеточия
        IFS=:
        set $one
        # Если $2 установлен, это значит, что значение в коде
        # ASCII и может быть записано в выходной файл
        # напрямую, если это не dn
        write_when_not_dn "$2" $1 1
        # Если $3 установлен, значит, что значение в коде
        # base64 и должно быть преобразовано перед записью,
        # если это не dn
        write_when_not_dn "$3" $1 0
    done
}

```



```
# Если же одна переменная - это промежуточная строка,
# и она должна быть присоединена к значению буфера
if [ ${#2} -eq 0 ] && [ ${#3} -eq 0 ] && [
    [ $skip -eq 0 ]; then
    _tempstr=$_tempstr$1
fi
IFS=$newline
done
# Если данные не требуют преобразования - выводим
# в один файл, если требуют - в другой, потому что
# mmencode пытается преобразовать все данные,
# переданные ей
if [ $enconv -eq 1 ]; then
    echo $_tempstr >> unconv.data
else
    echo $_tempstr >> conv.data
fi
# Сортируем данные в обоих файлах и удаляем
# дублирующие строки
cat unconv.data | sort -df | uniq > $outfile
cat conv.data | sort -df | uniq > deconv.data
# Построчно перекодируем файл
deconv=`cat deconv.data`
for one in $deconv
do
    echo $one | $util mmencode -u | $util iconv -
        -f utf-8 -t koi8-r >> $outfile
done
IFS=$saveifs
```

Первые три строки после объявления переменных – пример того, как можно отбросить путь (еще для этого используется \$(basename \$1)) и расширение файла, оставив только его имя. Затем отбрасываем в файле все комментарии и пустые строки (unspace()) и начинаем его построчную разборку. Поскольку в выводе ldapsearch имя поля отделяется от данных двоеточием, каждую строку разбиваем по этому символу.

Если после разбора есть второе поле, то данные не нужно преобразовывать, достаточно проверить, не dn ли это (write_when_not_dn()). Если же после разбора есть третье поле (второе при этом не проверяется), то значит, это base64-данные, и они должны быть преобразованы, опять же, если это не dn (write_when_not_dn()). Если же нет ни второго, ни третьего полей – это промежуточная строка закодированных данных, и она должна быть присоединена к буферу, в которые помещается недописанная строка. Это продолжается до конца входного файла.

Если были обнаружены закодированные данные, то сбросить последнюю строку данных в файл с данными для декодирования, иначе в файл, который не будет декодирован. Дело в том, что mmencode не проверяет переданный ей текст, а пытается перекодировать его, независимо от содержимого.

После этого отсортировать файлы, отобрать уникальный текст (зачем перекодировать что-то более одного раза?) и декодировать файл с закодированным текстом. Декодирование идет по одной строчке с последующим преобразованием в KOI8 (это можно убрать, кому не нужно) и выводом символа перевода строки после каждой строки.

Последней интересной процедурой является write_when_not_dn(). Она не очень большая, но весьма запутанная, потому что постоянно используются внешние переменные:

```
# Сформировать файл с данными для преобразования, если оно
# требуется, иначе с данными, не требующими преобразования,
# в том случае, если это не dn
# Выход: $1 line (строка) - данные из исходного файла
```

```
# $2 attribute (строка) - имя атрибута
# $3 enconv (digit) - 1 значит, что данные
# не требуют преобразования
# Выход: none
write_when_not_dn()
{
    # Если установлен $1, значит, это первая строка данных,
    # и нужно записать предыдущее содержимое
    # промежуточного буфера
    [ ${#1} -eq 0 ] && return
    # Если промежуточный буфер содержит накопленные данные,
    # их нужно записать
    if [ ${#_tempstr} -ne 0 ] && [ $skip -ne 1 ]; then
        # Если данные не требуют преобразования, записать
        # в один файл, если требуют - в другой, потому что
        # mmencode преобразовывает все
        if [ $enconv -eq 1 ]; then
            echo $_tempstr >> unconv.data
            enconv=0
        else
            echo $_tempstr >> conv.data
        fi
    fi
    # Если атрибут - dn, мы пропускаем эту и следующие строки
    # до тех пор, пока не начнется
    # новый атрибут
    if [ $2 = "dn" ]; then
        # Пропустить следующие строки
        _skip=1
        return
    else
        # Использовать следующие строки и накапливать данные
        # во временном буфере
        _skip=0
        _tempstr=$1
        # Отметим, если данные не требуют преобразования
        if [ $3 -eq 1 ]; then
            _enconv=1
        fi
    fi
}
```

Если первый параметр не задан (а такое сплошь и рядом, когда разбираются закодированные данные), то сразу выйти, ничего не делать. Если существуют данные во временном буфере, сброшенные туда при предыдущем вызове, то их нужно записать в соответствующий файл.

Если атрибут – dn, то пропустить данные, переданные при вызове, и все следующие строки, пока не будет снова обнаружено два поля в строке. Иначе – поместить данные во временный буфер, и, если они не требуют преобразования, отметить это.

Заключение

Язык Bourne Shell не очень-то годится для таких вещей, как обработка данных, полученных из AD, честно говоря. Приходится пускаться на различные ухищрения, и все равно работа получается с ограничениями – например, мне так и не удалось сделать одновременный запрос нескольких полей. Но для решения несложных задач типа приведенной его вполне можно использовать.

1. С. Реймер, М. Малкер. Active Directory для Windows 2003 Server. Справочник администратора. Пер. с англ. – М.: «СП-ЭКОМ», 2004 – 512 с.
2. Active Directory, 2nd Edition. Robbie Allen, Alistair G. Lowe-Norris. O'Reilly, 2003. ISBN 0-596-00466-4.
3. Ч. Расселл, Ш. Кроуфорд, Дж. Деренд. Microsoft Windows 2003 Server. Справочник администратора/Пер. с англ. – М.: Издательство «СП ЭКОМ», 2004 – 1392 с.: ил. ISBN 5-9570-0016-7.
4. <http://openoffice.mirahost.ru> – Сайт, где выложен скрипт.
5. <http://www.ietf.org/rfc/rfc2254.txt>.
6. <http://www.ietf.org/rfc/rfc2849.txt>.

Панель управления хостингом ISPConfig



Сергей Яремчук

Управление хостингом никогда не являлось простой задачей. Здесь на помощь приходят специальные инструменты, называемые панелями управления хостингом. Одним из таких решений является ISPConfig.

В журнале уже шла речь об одной из панелей управления хостингом — SysCP [1]. Недавнее знакомство с этим продуктом показало, что за прошедший год мало что изменилось, номер версии далеко вперед не ушел (сегодня актуальна 1.2.19). Но анализ ссылок по запросу «hosting» на сайте freshmeat.net показал, что расклад среди свободных решений стал несколько иной. Из всего списка постоянно обновляются всего десяток проектов. На момент написания этих строк проблем с локализацией не было в SysCP, DTC, VHCS, ispCP. Проект VHCS (Virtual Hosting Control Panel) [6] уже практически два года не обновлялся и содержит несколько критических ошибок. Учитывая, что некоторые разработчики на его основе создали ispCP, использовать VHCS не стоит.

Есть средства для локализации и в ISPConfig [2], о котором речь пойдет дальше, но в последней стабильной версии 2.2.25 этому вопросу приходится уделять несколько больше времени, чем в остальных продуктах. Тем более что здесь небольшая путаница. В описании на сайте сказано, что русский язык поддерживается изначально,

но, в процессе установки и работы он не предлагается вообще (см. рис. 1), хотя языковые файлы присутствуют (не для всех меню). Отдельно файлы локализации можно скачать по адресу [3]. В версии 3, находящейся в стадии активной разработки, русский интерфейс пока недоступен. Но система имеет удобный интерфейс для его локализации, и при знании базового английского, потратив несколько часов, легко это сделать самостоятельно. Готовый на 80% файл доступен на странице «Исходный код» сайта журнала [4] и отправлен авторам ISPConfig, по заверениям которых будет в ближайшем времени выложен на SVN.

В остальном функции проектов схожи, и, выбрав любой, можно:

- управлять веб-сервером (обычно Apache 1.x/2.x), создавать виртуальные серверы;
- редактировать DNS-записи (A, CNAME, MX, иногда SPF) на поддерживаемом DNS-сервере, создавать домены и субдомены;
- управлять базами и пользователями MySQL (другие почему-то мне не встречались, по крайней мере в списке основных);

- управлять виртуальными почтовыми серверами и ящиками пользователей;
- управлять почтовыми фильтрами, автоответчиками, антиспам- и антивирус-проверками, работать с почтой через веб-интерфейс;
- устанавливать дисковые квоты и лимиты трафика;
- управлять виртуальными FTP-серверами и пользователями;
- управлять SSL-сертификатами, использованием PHP и CGI;
- несколько уровней доступа в зависимости от задач: администратор сервера, реселлер, пользователь (ресурсов сервера). Некоторые решения предоставляют и возможность доступа к почтовым ящикам и обычным пользователям;
- собирать статистику по всем вопросам;
- отслеживать обращения в техподдержку (ticket system).

Все эти функции есть и ISPConfig. Отличия лишь в серверах, которые поддерживает та или иная панель. Если места для веб и СУБД традиционно заняты Apache и MySQL (альтернативы

для них предлагаются реже), то для организации SMTP/POP3/IMAP может быть задействовано любое из доступных сегодня свободных решений. Поэтому при выборе хостинг-панели следует внимательно присмотреться к требованиям. Но, например, Domain Technologie Control (DTC) [5], который на сегодня поддерживает самое большое количество серверов, из-за своей функциональности не так прост в установке. Не спасает и то, что он есть в репозиториях многих дистрибутивов (например, в отличие от остальных решений он единственный присутствует в Ubuntu).

Официально ISPConfig поддерживает все популярные сегодня дистрибутивы: Mandrake 8.1 – 10.2, Mandriva от 2006, Red Hat Linux 7.3 – 9.0, Fedora (Core) от 1, CentOS 4.1 до 5.2, SuSE Linux 7.2 – 11.0, OpenSuSE от 5.2, Debian от 3.0 и Ubuntu от 5.04. Хотя, вероятно, будет работать и в других решениях, для которых доступны:

- Apache 1.3.x или / 2.x;
- Sendmail или Postfix;
- ProFTPD и Pure-FTPd как (standalone) или vsftpd как (inetd/xinetd/standalone);
- PHP 4.0.5 и выше;
- MySQL 4/5;
- POP3/IMAP, поддерживающий форматы – традиционный Unix-Mailbox (gnu-pop3d, qpopper, ipop3d, popa3d или vmpop3d) или Maildir (Courier-IMAP, Dovecot);
- BIND 8/9 для ISPConfig 2 или MyDNS для ISPConfig 3;
- phpMyAdmin, Webalizer, Quota, iptables или ipchains.

И некоторые другие приложения. На странице загрузки можно найти ссылки на некоторые дополнительные инструменты, расширяющие стандартные возможности ISPConfig. Распространяется ISPConfig по BSD-лицензии.

Установка ISPConfig

Как уже говорилось, на данный момент развиваются две версии: стабильная 2.2.25 и тестовая 3.0.0.6 Beta. Их интерфейс, а также процесс установки и локализации отличаются. Например, в версии 3 для установки компонентов используются программы из репозитория дистрибутива, а в версии 2 некоторые приложения (Apache 1.3.41, ClamAV 0.93.3, PHP 5.2.6 и некоторые другие) компилируются из исходных текстов. Все программы, которые нужно компилировать, включены в архив, поэтому и размер его почти 45 Мб. Даже при соблюдении всех инструкций компиляция иногда заканчивается неудачей, и установочные скрипты по разным причинам (чаще несовместимость библиотек) отказываются дальше работать. Скрипты рассчитаны именно на такую установку, к ним привязаны и инструкции. Если, к примеру, необходим Apache2, процесс установки несколько усложняется.

Так как в будущем, вероятно, версия 3 будет основной, о ней и пойдет речь дальше. Хотя, сравнивая возможности интерфейса, версия 3 на данный момент проигрывает 2. Например, возможности реселлера пока реализованы не пол-

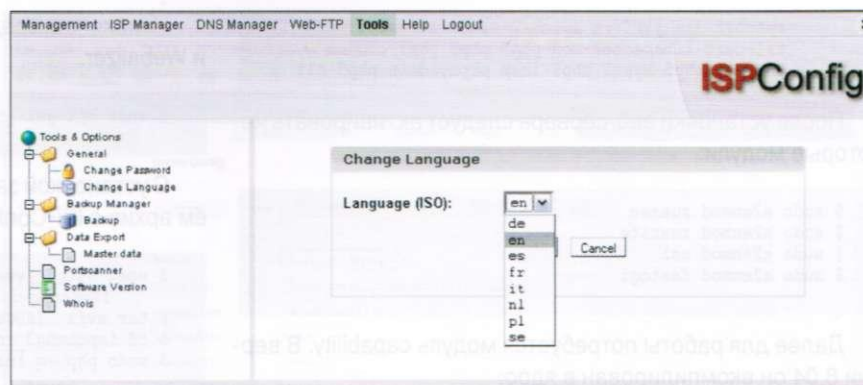


Рисунок 1. В интерфейсе ISPConfig русский язык не предлагается

ностью, поэтому грань между ним и пользователем практически отсутствует. Но для организации хостинга без реселлеров того, что есть, вполне достаточно. Также будем учитывать, что перед нами все-таки бета, и в дальнейшем все может измениться.

В качестве системы для установки рекомендованы Debian 4.0 или Ubuntu 7.10. Я использовал Ubuntu 8.04 LTS, проблем в работе замечено не было. Инструкции по установке найдете на сайте проекта и внутри архива.

Вначале обновляем систему:

```
$ sudo apt-get update
$ sudo apt-get -u upgrade
```

Приступаем к установке компонентов будущей системы, если какие-то из пакетов уже в системе присутствуют, их установку можно пропустить. Устанавливаем Postfix, Courier, Saslauthd, MySQL, phpMyAdmin:

```
$ sudo apt-get install postfix postfix-mysql mysql-client
mysql-server courier-authdaemon courier-authlib-mysql
courier-pop courier-pop-ssl courier-imap
courier-imap-ssl libsasl2-2 libsasl2-modules
libsasl2-modules-sql sasl2-bin libpam-mysql
openssl courier-maildrop getmail4
```

В процессе установки приложений мастер настройки пакетов будет задавать вопросы по конфигурации. При установке Courier на вопрос «Create directories for web-based administration?» отвечаем «No». При установке Postfix на вопрос «General type of configuration?» выбираем «Internet site». Далее вводим имя почтового домена и на вопрос «SSL certificate required» выбираем «Ok».

Пароль для пользователя root в MySQL по умолчанию отсутствует. Его нужно добавить.

```
$ mysqladmin -u root password password
```

Теперь очередь Amavisd-new, Spamassassin и Clamav:

```
$ sudo apt-get install amavisd-new spamassassin clamav
clamav-daemon zoo nomarch lzop cabextract
apt-listchanges libnet-ldap-perl
libauthen-sasl-perl daemon libio-string-perl
libio-socket-ssl-perl libnet-ident-perl
libnet-dns-perl
```

Далее Apache2, PHP5 и phpMyAdmin:

```
$ sudo apt-get install apache2 apache2.2-common
```


администрирование

```
apache2-mpm-prefork apache2-utils libexpat1 \
ssl-cert libapache2-mod-php5 php5 php5-common \
php5-gd php5-mysql php5-imap phpmyadmin php5-cli
```

После установки веб-сервера следует активировать некоторые модули:

```
$ sudo a2enmod suexec
$ sudo a2enmod rewrite
$ sudo a2enmod ssl
$ sudo a2enmod fastcgi
```

Далее для работы потребуется модуль `capability`. В версии 8.04 он вкомпилирован в ядро:

```
$ grep -i capabilities /boot/config-2.6.24-16-generic
CONFIG_SECURITY_CAPABILITIES=y
```

Иначе его следует загружать в виде модуля:

```
$ sudo modprobe capability
$ sudo echo 'capability' >> /etc/modules
```

Ставим пакеты `Pure-FTPd` и `quota`:

```
$ sudo apt-get install pure-ftpd-common \
pure-ftpd-mysql quota quotatool
```

Отключаем определение имен узлов в журнале `Pure-FTPd`, при этом записи станут менее информативными, но зато будет требоваться меньше ресурсов:

```
$ sudo sh -c 'echo 'yes' > /etc/pure-ftpd/conf/DontResolve'
```

Для третьей версии `ISPConfig` в качестве DNS-сервера рекомендуется `MyDNS`. На вопрос «Почему?» ответа не получил. Все стартовые скрипты рассчитаны именно на него, а не `BIND`. В репозитории `Ubuntu` `MyDNS` нет, поэтому его нужно скомпилировать. Ставим пакеты, необходимые для сборки:

```
$ sudo apt-get install build-essential libmysqlclient15-dev
```

Скачиваем и устанавливаем `MyDNS` обычным образом:

```
$ wget http://mydns.bboy.net/download/mydns-1.1.0.tar.gz
$ tar xvfz mydns-1.1.0.tar.gz
$ cd mydns-1.1.0
$ ./configure
$ make
$ sudo make install
```

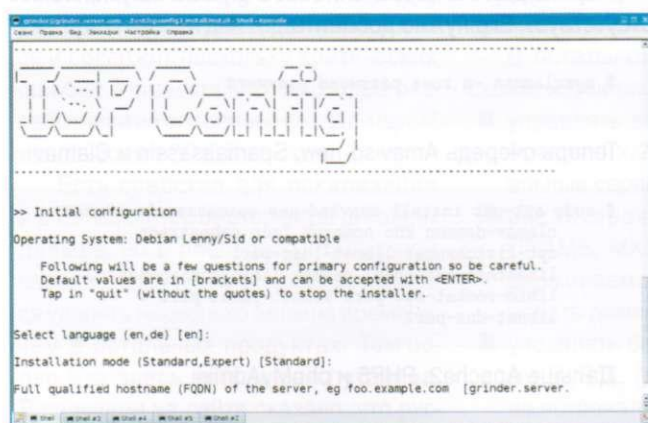


Рисунок 2. Работа установочного скрипта

И, наконец, средства анализа файлов журналов `vlogger` и `Webalizer`.

```
$ sudo apt-get install vlogger webalizer
```

С установкой зависимостей закончили. Теперь скачиваем архив с `ISPConfig`, распаковываем и ставим:

```
$ wget http://www.ispconfig.org/downloads/ \
ISPConfig-3.0.0.6-beta.tar.gz
$ tar xvfz ISPConfig-3.0.0.6-beta.tar.gz
$ cd ispconfig3_install/install/
$ sudo php -q install.php
```

После запуска установочный скрипт будет задавать ряд вопросов (см. рис. 2). Здесь ничего необычного. В квадратных скобках предлагаются ответы по умолчанию, если он не подходит, вводим свое значение. После установки информацию можно найти в журнале `/var/log/ispconfig_install.log`:

- `Select language (en,de) [en]:` – выбираем язык;
- `Installation mode (Standard,Expert) [Standard]:` – вариант установки.

В варианте `Expert` скрипт будет задавать чуть больше вопросов по настройке конкретного сервиса, обычно необходимости прибегать к нему нет.

- `Full qualified hostname (FQDN) of the server, eg foo.example.com [example.com]:` – вводим полное имя сервера;
- `MySQL server hostname [localhost]:` – параметры для доступа к `MySQL`.

Если `MySQL` принимает подключения не на стандартном порту, придется подправить команду подключения внутри установочного скрипта.

- `MySQL root username [root]:` – имя `root`;
- `MySQL root password []:password` – пароль для `root`;
- `MySQL database to create [dbispconfig]:` – создание базы данных.

Далее скрипт создаст базы данных и таблиц, и будет сгенерирован сертификат, в процессе создания которого будут заданы традиционные вопросы (страна, город, организация и так далее) и предложено указать порт для подключения к `ISPConfig`. По умолчанию – 8080, если этот порт уже занят, меняем его, введя новое значение.

Затем будут созданы пользователь и группа `ispconfig`, установлены задания для `crontab` и запущены все сервисы.

Веб-интерфейс ISPConfig

После установки набираем в браузере `http://hostname:8080`, для регистрации используем учетную запись – `admin` и пароль – `admin` (пароль после регистрации не забываем сменить). Интерфейс визуально разделен на три поля и стандартен для такого рода программ (см. рис. 3). Вверху находится основное меню, открывающее доступ к определенным настройкам: `System`, `Client`, `Email`, `Monitor`, `Sites`, `DNS`. Плюс по умолчанию отключенный модуль `BE Designer` (включается в настройках учетной записи). При выборе любого пункта слева будет доступна группа меню, относящаяся к нему. Все настройки производятся в центре страни-

цы. Чтобы разобраться с возможностями интерфейса, достаточно потратить полчаса времени, все находится на своих местах и там, где ожидаешь. Сюрпризов не возникает. На создание новой DNS-записи, домена, субдомена, виртуального почтового или веб-сервера, почтового ящика и прочего уходит 1-2 минуты. Заполняем предложенные поля и следуем указаниям системы, в случае проблем появится предупреждающая надпись.

При создании учетной записи нового пользователя доступные ему пункты меню реализуются при помощи подключения модулей. Последняя функция очень удобна, так как пользователю можно дать именно такой интерфейс, который полностью подходит для выбранной задачи. Ни больше ни меньше. Например, можно создать несколько записей с правами администратора, передав им часть функций по управлению. Меню BE Designer позволяет редактировать меню, убирать или добавлять новые пункты. При создании клиентской учетной записи есть возможность установки разнообразных лимитов.

Интерфейс версии 3 переведен на 6 языков, русского в этом списке нет. Но разработчики предоставили удобный инструмент, чтобы сделать это самостоятельно.

В меню System находится отдельный пункт Language Editor. Есть, как минимум, два варианта работы с ним. Для начала создаем новый язык, выбрав New Language, затем в поле Select language basis указываем язык, используемый в качестве базового, и вводим название из двух букв, в New language (в нашем случае ru). Теперь, перейдя в меню Languages, выбираем в списке Select language свой язык и получаем возможность редактирования отдельных файлов при помощи веб-интерфейса (см. рис. 4). Но удобнее экспортировать языковой файл, выбрав в меню Export язык. Созданный файл сохраняем на жесткий диск, в дальнейшем его можно редактировать при помощи любого текстового редактора, поддерживающего юникод. По окончании импортируем его через меню Import.

Несмотря на то что перед нами пока еще версия ISPConfig, находящаяся в разработке, она полностью справ-

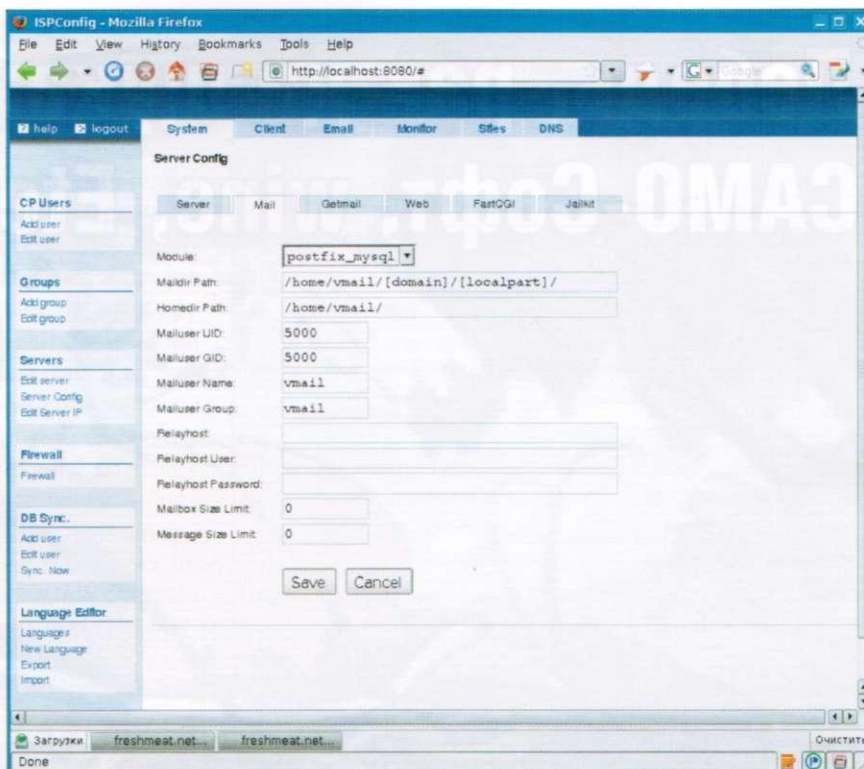


Рисунок 3. Интерфейс администратора ISPConfig

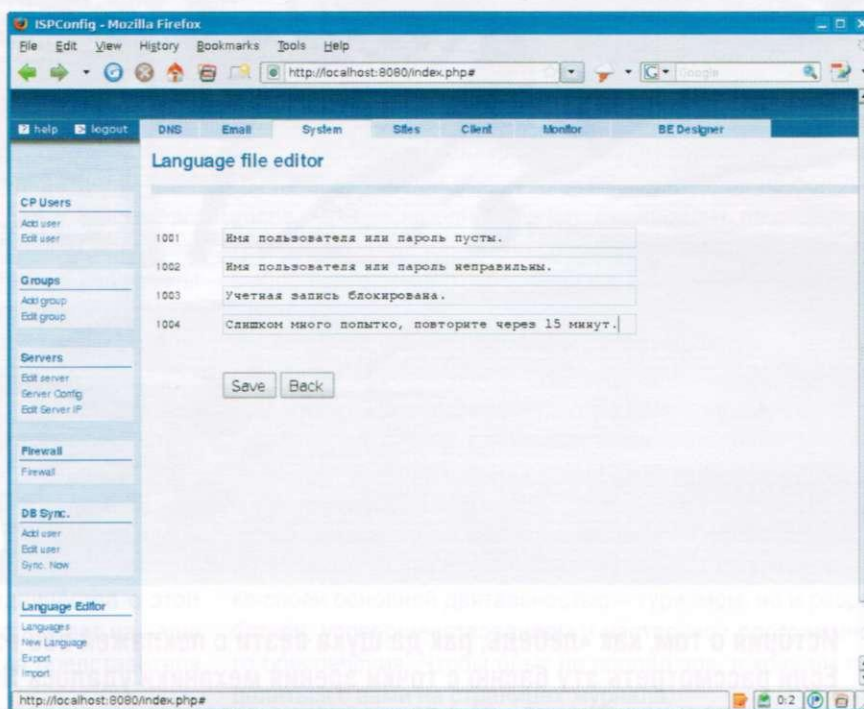


Рисунок 4. Редактирование языкового файла

ляется со своими задачами. Если часто необходимо редактировать DNS-записи, создавать виртуальные серверы, стоит присмотреться к этому решению.

1. Яремчук С. Панель управления хостингом SysCP. // Системный администратор, №8, 2007 г. — С. 74-79.

2. Сайт проекта ISPConfig — <http://www.ispconfig.org>.
3. Файл локализации для версии 2.x — <http://www.wifi-ufa.ru/ispconfig.ru.lng.zip>.
4. <http://www.samag.ru/source>.
5. Сайт проекта Domain Technologie Control — <http://www.gplhost.com/software-dtc.html>.
6. Сайт проекта VHCS — <http://vhcs.net>.

Лебедь, рак да щука: САМО-Софт, wine, Etersoft



Сергей Барановский

История о том, как «лебедь, рак да щука везти с поклажей воз взялись», известна всем. Если рассмотреть эту басню с точки зрения механики удалось Якову Перельману в своей книге «Занимательная физика», то с точки зрения автоматизации работы турфирм ситуация очень похожа и можно назвать аналогичную тройцу: «САМО-Софт», wine, Etersoft.

Общее положение турфирм на рынке автоматизации их деятельности было описано в майском номере [2], как и решение по запуску «САМО-Тур» под Linux. Дальше описываются произошедшие изменения после выхода статьи, а также рассматривается ещё один туристический

продукт — «САМО-ТурАгент» и описывается способ его запуска под Linux.

«Жареный петух клюнул»

Результат от моей первой статьи превзошёл ожидания. Если вопросы вида «А твоя ли эта статья?» были ожидаемы, то другие вещи заставили ме-

ня задуматься. Например, оказалось, что тираж журнала велик, продукт более чем известен и интересен читателям, а территория распространения журнала широка.

Так, при личном общении главный разработчик фирмы «САМО» признался, что «ксерокс» статьи ему подарили

коллеги сразу после выхода журнала в свет, когда он был в командировке в Украине.

Ошибка «Variant does not reference an automation object.» была исправлена уже в июньском выпуске «САМО-Тура», а вот сообщить нам об этом и что делать дальше «САМО» так и не смогли. Самое главное, что дело сдвинулось с «мёртвой точки» и начался диалог. Нами был передан на временное хранение в фирму «САМО» компьютер с установленным Linux для «экспериментов на месте», а так как наша фирма использует в своей деятельности ещё и другой продукт фирмы «САМО», о котором речь пойдёт в этой статье, то общение шло по двум направлениям. Так, программистом из «САМО» Игорем Калашниковым мы были направлены в фирму Etersoft [3], которая как раз специализируется на запуске различных приложений под Linux, а другой программист – Куракин Вячеслав исправил ряд ошибок в работе «САМО-ТурАгента». Всё это положительно сказалось на автоматизации нашей турфирмы, но, несмотря на это, оперативность, инициативность и качество работы с клиентами оставляют желать лучшего.

Etersoft

Если зайти на сайт компании [3], то в разделе «о фирме» можно узнать, что «компания «Этерсофт» была создана в конце 2003 года группой Linux-разработчиков для оказания услуг по разработке решений на базе операционной системы ALT Linux; одной из миссий является популяризация и внедрение свободных программ в любых областях, где применяется компьютерная техника».

Фирма выпускает много интересных решений по разумным ценам и бесплатных. Вот некоторые из них, что привлекли наше внимание: модифицированная и улучшенная версия wine, универсальный транслятор SQL-запросов из диалекта T-SQL в pgSQL (он позволяет использовать PostgreSQL в приложениях, разработанных для работы с MS SQL), UniOffice (позволяет использовать Open Office.org вместо MS Office, выполняя трансляцию COM-запросов к MS Office в вызовы Open Office.org).

Что интересно, фирма открыта к сотрудничеству и без труда предоставляет коммерческие версии своих продуктов для тестирования по «гарантийному письму» в их адрес, чем мы и воспользовались.

Общение с техподдержкой, как для клиентов, так и для «тестеров», происходит по электронной почте.

Договориться об иной форме сотрудничества с этой фирмой у нас, увы, не получилось, даже в ответ на наше предложение оплатить командировку для представителя Etersoft в наш офис.

Лебедь, рак да щука

Нельзя сказать, что «воз и ныне там», но ситуация очень похожа. Несмотря на столь радужные перспективы, открытые журналом в мае, «русский сервис» продолжает быть ненавязчивым. Как фирма «САМО» вырала себе большую яму и не справляется с валом запросов от клиентов на фоне текучести кадров, так и фирма «Этерсофт» не успела достаточно вырасти, чтобы понять, насколько перспективным может быть рынок по автоматизации работы турфирм. При этом месяцы общения с обеими фирмами показали,

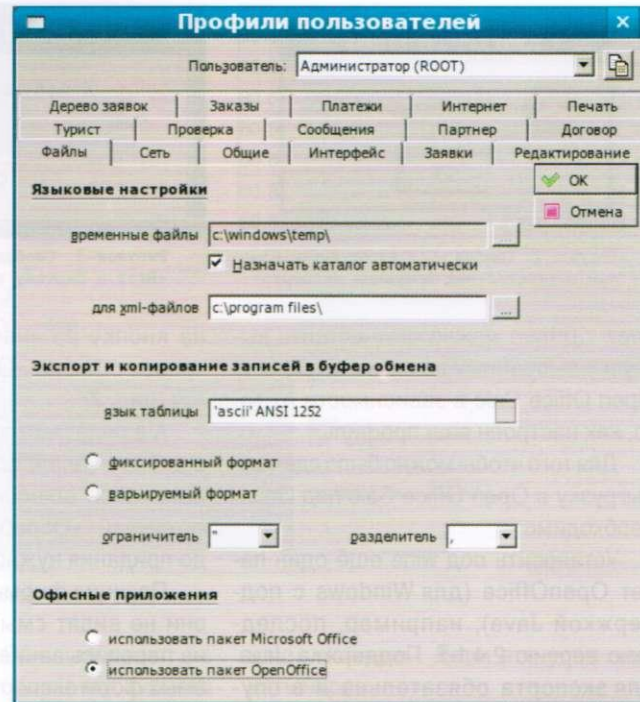


Рисунок 1. Переключение на OpenOffice в профиле пользователя

что они отнюдь не заинтересованы в совместных проектах, хотя их координаты неоднократно пересылались друг другу и доступны на сайтах фирм.

Скорее всего, эта ситуация связана с недостаточно сформированным спросом на услуги со стороны турфирм и с тем, что у производителей программного обеспечения нет существенных конкурентов. Многие маленькие турфирмы по всей России имеют число компьютеров не больше числа пальцев на одной руке и, как вариант, не содержат технических отделов, т.е. не способны обоснованно сделать выбор программного обеспечения, а также имеют затруднения с настройкой.

В случае каких-либо проверок по используемому программному обеспечению турагентствам проще переплатить и купить понятный им Windows, чем связываться с отечественными разработчиками программного обеспечения. То есть вместо того, чтобы кормить свою армию программистов, мы кормим чужую.

«Под лежащий камень вода не течёт» – гласит пословица. Поэтому наша фирма вынуждена заниматься не только своей основной деятельностью – туризмом, но и разработкой, усовершенствованием и настройкой программного обеспечения. Чтобы опыт не пропал зря, я хочу им поделиться с вами на страницах журнала.

Экспорт из «САМО-Тура»

При запуске «САМО-Тур» под Linux ранее [2] имелась проблема экспорта. На сегодня есть несколько решений. Первое из них сделано средствами «САМО-Тура», через xml-файл, второе – с использованием транслятора COM-объектов.

Xmlfile

Если в любом из выводимых списков или таблице выделить несколько строк, например, пробелом, то можно вызвать правой кнопкой мыши меню, где будет пункт «экс-

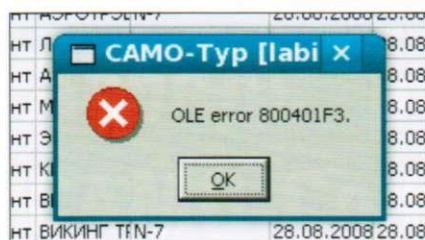


Рисунок 2. Ошибка «OLE error 800401F3» при осуществлении операции экспорта

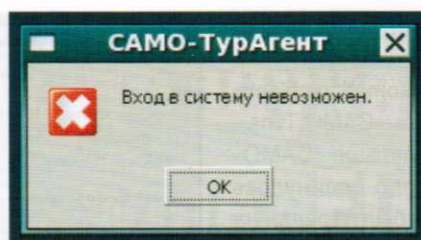


Рисунок 3. Сообщение об ошибке «Вход в систему невозможен»

порт». Далее можно осуществить выгрузку выбранных данных в Excel или Open Office Calc в зависимости от того, как настроен ваш профиль.

Для того чтобы можно было сделать выгрузку в Open Office Calc под Linux, необходимо:

Установить под wine ещё один пакет OpenOffice (для Windows с поддержкой Java), например, последнюю версию 2.4.1-1. Поддержка Java для экспорта обязательна и в случае работы под Windows тоже нужна. Иначе при попытке экспорта будет выдана ошибка «Для выполнения данного задания OpenOffice.org необходимо установленное окружение Java(JRE). Установите JRE и перезапустите OpenOffice.org.». Использование двух пакетов OpenOffice (родной под Linux и второй под wine) можно отнести к издержкам данного способа и надеяться, что когда-нибудь программа будет одна.

Переключить в профиле пользователя «Использовать пакет OpenOffice» (см. рис 1).

Под wine запустить редактор реестра (wine regedit), далее в нём перейти в следующий раздел HKEY_CLASSES_ROOT\xmlfile\shell\open\command и исправить значение по умолчанию на «"путь_к_openoffice" %1». Например:

```
"C:\Program Files\OpenOffice.org 2.4\program\soffice.exe" %1
```

В половине используемых на практике случаев этого хватает. Но другая половина экспорта остаётся так и недоделанной со стороны фирмы «САМО». Проблема экспорта у них существует не только под Linux. Если взять компьютер, где будет Windows, «САМО-Тур» и OpenOffice и не будет MSOffice (т.е. исключаем проблемы портирования под Linux), то в ряде мест экспорт всё равно не работает. Естественно, все эти проблемы наследуются при запуске под wine. Так, если нажать

на кнопку со значком Excel, выскакивает ошибка «OLE error 800401F3» (см. рис. 2).

А в ряде мест так и не исправлена ошибка «Variant does not reference an automation object.». Очень напоминает случай: «Обработать напильником до придания нужной формы».

Позиция фирмы «САМО» понятна, они не видят смысла тратить время на переписывание нечасто используемых форм экспорта, когда нет времени и во всю идёт разработка «САМО2», а также появилась возможность вывода форм через «FastReport» [5] и «QuickReport».

FastReport

Хочется отметить, что «FastReport» [5] – это интересный проект, заслуживающий отдельного внимания. Он удобен тем, что без программирования позволяет делать вывод всевозможных форм и отчётов, вставляя в них различную информацию, выдаваемую программами. Несколько слов о настройке wine на работу с «FastReport» будет сказано в разделе «САМО-ТурАгент». Всё упомянутое дальше можно смело использовать при работе с «САМО-Туром» и тем самым увеличить число мест в программе, откуда можно делать вывод информации.

Uni Office

Вторым из решений может быть использование транслятора COM-объектов [4]. Для этого следует скачать с сайта [4] дистрибутив unioffice.msi и установить его командой:

```
$ wine msixec /i /путь/unioffice.msi
```

По факту установка данного пакета не сильно повлияла на экспорт. Основные необходимые функции экспорта продолжают не работать. Плюс в том, что этот продукт бесплатный и может быть полезен кому-либо ещё.

Также на сайте можно найти ссылки на похожие проекты, позволяющие программам вроде «1С» успешно работать с OpenOffice.

Другие решения

Существует достаточно много решений по запуску бухгалтерских программ «1С» под Linux. Один из способов – использование различных библиотек вместе с wine (например dcom98). Скорее всего, этим опытом можно воспользоваться и для продуктов «САМО». По крайней мере, форумы изобилуют различной информацией, на переработку и проверку которой требуется время. Также не следует забывать и про другие существующие эмуляторы вроде CrossOver Office. В любом случае решение будет найдено.

Экспорт в почтовую систему

Кроме экспорта различных таблиц, связь «САМО-Тура» с внешним миром происходит и через электронную почту. Сотрудники туроператорских офисов регулярно отправляют письма по электронной почте как своим партнёрам с просьбой забронировать ту или иную услугу, так и агентствам в виде писем-подтверждений.

По мере увеличения объёмов продаж создание новых писем и нахождение нужного адреса занимает много времени, поэтому требуется автоматизация этого процесса. Время сокращается за счёт использования различных шаблонов писем и экспорта в почту. Путём одного-двух кликов менеджеры получают готовое письмо, которое остаётся только прочитать и нажать кнопку «Отправить». В качестве почтового клиента нами используется Mozilla Thunderbird. Несмотря на то что этот клиент является кроссплатформенным, использование под Linux имеет свои тонкости. Родная Linux-версия не подойдёт, придётся ставить ещё одну под wine, ситуация схожа с OpenOffice. Далее, в Windows передачей данных от «САМО-Тура» к Thunderbird занимается библиотека mapi32.dll – её надо скопировать. Есть и 16-битная версия этой библиотеки – mapi.dll. Чтобы читатель не наступал на грабли, отмечу, что, используя Dependency Walker [6], можно вы-

яснить зависимости и получить имена ещё шести библиотек: advapi32.dll, gdi32.dll, kernel32.dll, ntdll.dll, rpcrt4.dll, user32.dll, которые, казалось бы, тоже нужны, но на практике установка всех их под wine приводит к ошибкам и неработоспособности приложений. Изначально имеющиеся wine-аналоги в директории system32 работают лучше, поэтому мы ограничимся копированием файлов mapi32.dll и mapi.dll в директорию ~/.wine/drive_c/windows/system32. Скорее всего и 16-битная версия не нужна.

После копирования следует сообщить wine о библиотеке, для этого через меню «Приложения → Wine → Wine Configuration» нужно запустить настройщик (см. рис. 7), где следует выбрать вкладку «Библиотеки» и там в падающем меню «Новое замещение для библиотеки» найти по одной (или написать) названия библиотек, после нажать кнопку «Добавить» и увидеть, как они добавятся в окошечке ниже.

После необходимо в реестре wine (меню «Приложения → Wine → regedit») создать раздел HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Messaging Subsystem и прописать в нём значение параметра «MAPI»=«1».

После этих операций у нас местами заработал экспорт в почту, например, если из меню «Справочники → Заявки» выбрать заявку с гостиницей и нажать на правом меню «E-mail → Бронирование → Для партнёра гостиницы ...», то там работает. А вот соседний пункт «E-mail → Подтверждение/Отмена → ...» выдаёт сообщение «Ошибка при отправке электронной почты». Исправить пока не удалось.

Более правильная позиция — это не работать с различными трансляторами запросов и конверторами, проводя рабочие часы на форумах, а чтобы разработчики программного обеспечения учитывали потребности рынка более гибко и сами создавали необходимые функции в своих продуктах. Никто, кроме самих разработчиков, не знает лучше свой собственный код, но по жизни получается, что мир держится на Кулибиных.

САМО-ТурАгент

Все туристические фирмы можно грубо поделить на две части по сферам

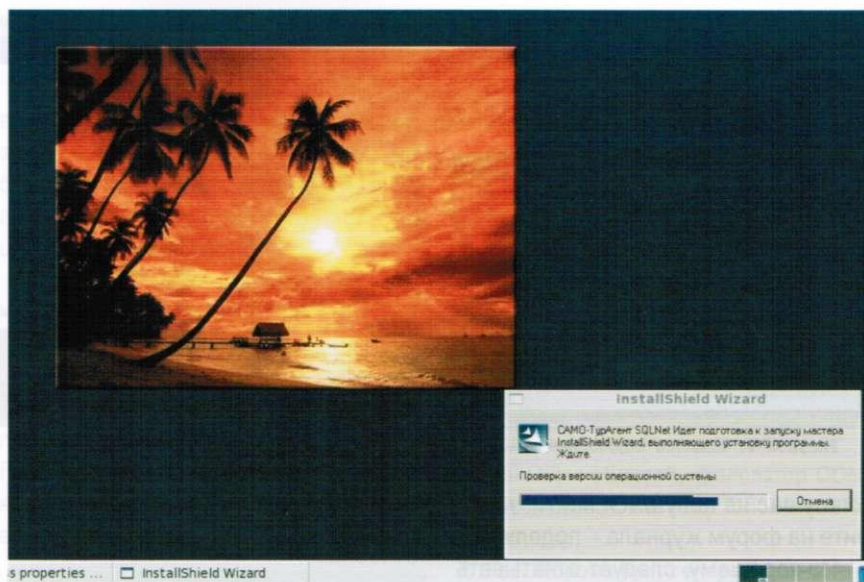


Рисунок 4. Установка «САМО-ТурАгента»

их деятельности. Одни работают непосредственно с будущими туристами — они называются турагентствами, а другие не работают с туристами напрямую, но обеспечивают сами услуги — это туроператоры.

Если продукт CAMO-Тур нужен операторам, то для автоматизации работы агентств существует другой продукт — это «САМО-ТурАгент».

Агентств больше, и, наверно, более правильным было бы первую статью посвятить именно этой программе, но т.к. её запуск значительно проще запуска «САМО-Тура», то эта программа идёт вторым планом.

Имея за плечами опыт использования этой программы под Linux во всех наших офисах продаж уже около года, хотелось бы рассказать о том, как заставить работать этот продукт под Linux, а также о тех недостатках, с которыми приходилось и приходится сталкиваться в работе.

Допускаю наличие более удобных продуктов других фирм, и было бы хорошо прочитать о них на страницах журнала, но по ряду причин наша фирма не смогла найти альтернативный продукт и мы вынуждены были купить этот.

В установке «САМО-ТурАгента» нет ничего сложного. Для этого требуется дистрибутив программы, установленный Linux, например Fedora 8, а также пакет wine. Если вы прочитали первую часть статьи [2], то проделать эти шаги для вас не составит труда.

А вот с wine нас ожидают неболь-

шие проблемы. Если его устанавливать через:

```
# yum install wine
```

то установится его последняя версия 1.0.1, а она, увы, не «дружит» с «САМО-ТурАгентом». Всё ставится, запускается, выдаётся окно запроса логина и пароля на вход, но после подключения не происходит и выдаётся сообщение об ошибке «Вход в систему невозможен» (см. рис. 3). Проблема решается установкой версии 0.9.58-1.fc8. Выяснилось это случайно, т.к. на момент начала работы с программой это была самая последняя версия wine и всё работало. После вышло обновление, с которым и возникли проблемы. Разработчики «САМО» подтвердили ситуацию, да и вообще на версию 1.0.1 много нареканий, поэтому пока единственное доступное решение сделать downgrade до версии 0.9.58. На других версиях проверка не производилась.

Найти старые версии в виде rpm можно, например, на <http://rpmfind.net>.

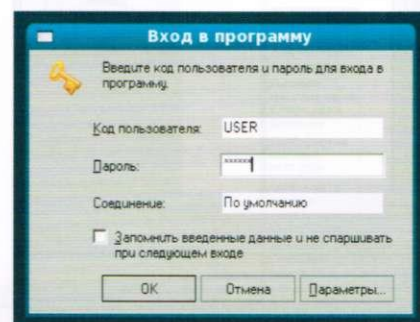


Рисунок 5. Вводим логин и пароль

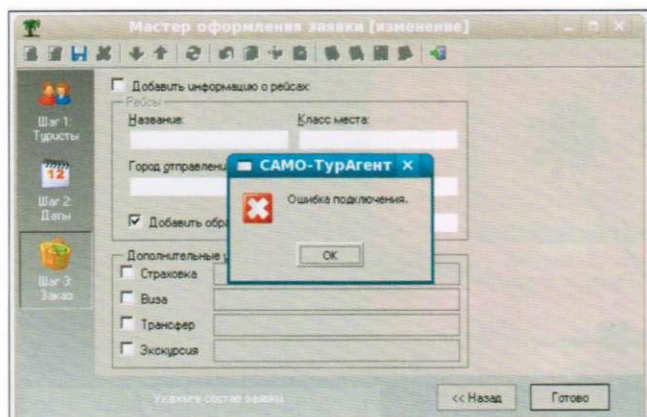


Рисунок 6. «Ошибка подключения»

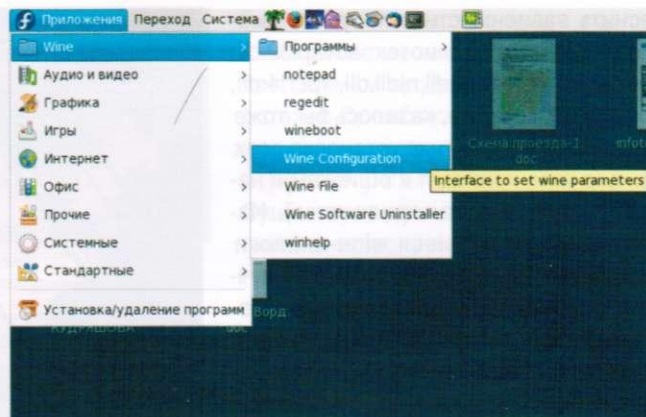


Рисунок 7. Wine Configuration

Если у вас не получится найти, то пишите на форум журнала – поделюсь.

По-хорошему, следует откатывать все файлы, но у меня работает и для двух основных. Сначала удаляем файлы с ключом --nodeps:

```
# rpm -e wine wine-core --nodeps
```

а затем устанавливаем предыдущие:

```
# rpm -ihv wine-0.9.58-1.fc8.i386.rpm \
wine-core-0.9.58-1.fc8.i386.rpm \
--nodeps --force
```

Далее копируем дистрибутив «СМО-ТурАгента» на компьютер и запускаем из-под wine файл setup.exe, после чего начинается обычная установка (см. рис. 4).

Далее, копируем с Windows-машины файл client.conf в директорию: ~/.wine/drive_c/Program Files/SAMO-Soft/TourAgent SQLNet 5.2 или создаём его

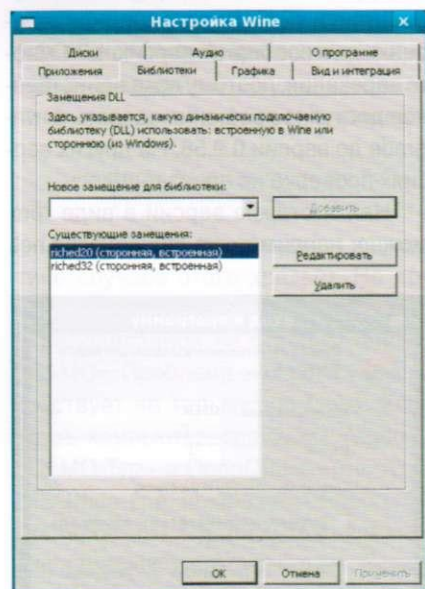


Рисунок 8. Настройка wine, замещения dll

сами примерно со следующим содержанием:

```
[Settings]^M
Locale=0^M
ConnectionIndex=0^M
^M
[Connection0]^M
Port=88^M
Name=Connection^M
Host=192.168.0.1^M
UserName=^M
Login=^M
```

где ^M – так отображается перевод строки. Не забывайте, что в Windows и Linux есть различия. Поэтому или по другим причинам с созданием файла автоматически, при запуске программы, возможны проблемы. При копировании работающего файла проблем нет.

Далее, запускаем программу, щёлкнув по иконке (пальмочка в меню «Приложения → Wine → Программы → CAMO-ТурАгент SQL 5.2 → CAMO-ТурАгент SQL 5.2»). Вводим логин и пароль и подключаемся (см. рис. 5).

При первой установке будет запрос от Wine Gecko Installer – нажимаем Install. В случае отмены – нажатие Cancel, вопрос будет задан при повторном запуске, а в правом нижнем углу не будет показана информационная панель agentinfo.

Информация на панели берётся с адреса <http://www.samo.ru/agentinfo/>, и кажется, в настройках программы этот адрес не меняется. Если в вашем случае для выхода во внешний мир используется прокси-сервер с аутентификацией, то программа не предоставляет отдельного меню по настройке его параметров.

Что интересно, в первых версиях «СМО-ТурАгента» при работе из локальной сети версий, запущенных

под Windows, если в настройках IE был прописан прокси, то и программа подключалась, используя этот прокси. При этом галочка «Использовать прокси-сервер» установлена не была. Приходилось её ставить, прописывать «левый» прокси, а после в списке исключений прописывать реальный адрес сервера, тогда подключение к нему шло напрямую.

Также интересно, что настройка для поля agentinfo находится отдельно от настроек подключения. Так разработчики нам сообщили, что сама программа при подключении к серверу не обращает внимания на системные настройки прокси у IE, а вот agentinfo (справа внизу) на них смотреть будет. Настройки лежат в реестре wine. Проверьте в ветке реестра HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings параметр ProxyEnable, он должен быть установлен в 1. ProxyServer хранит адрес прокси-сервера, например server:3128. Параметры аутентификации, скорее всего, следует прописать там же.

На мой взгляд, это не самое удачное решение, но другого нет.

Что ещё не работает (не работало)?

Первые версии программы не закрывались с помощью «крестика» в правом верхнем углу. Пришлось обучить всех пользователей использовать xkill. Далее, в середине лета Вячеславом Куракиным ошибка была исправлена. Размер файла client_sqlnet.exe, где ошибка исправлена, равен 5710336 байт.

При сворачивании окна оно практически никогда после не разворачи-

вается, и программу приходится завершать и запускать заново. Для этого пользователи были научены работать с разными виртуальными столами и запускают «САМО-ТурАгент» на отдельном столе.

При попытке сохранить данные из «Мастера оформления заявки» выдаётся окно «Ошибка подключения» (см. рис. 6). При этом данные всё же сохраняются. Аналогичная ошибка выскакивает при удалении заявки. Пользователи просто игнорируют ошибку и работают, как если бы её не было.

При печати отчётов используется «FastReport» [5]. Была ситуация, что не видны часть шрифтов, пропадают буквы и абзацы текста. Решение было найдено следующее – скопировать с Windows машины в папку `~/.wine/drive_c/windows/system32` файлы `riched20.dll` и `riched32.dll`. После этого надо их прописать в конфигурации

wine (меню «Приложения → Wine → Wine Configuration») (см. рис. 7).

Во вкладке «Библиотеки» следует выбрать в падающем меню «Новое замещение для библиотеки» нужные файлы и нажать после кнопку «Добавить», после чего они появятся в окошечке ниже (см. рис. 8).

После следует скопировать нужные шрифты в папку `~/.wine/drive_c/windows/fonts`. Например, можно попробовать взять целиком папку `Fonts` с какой-нибудь Windows-машины. После этих действий все отчёты через FastReport отображаются, и их можно печатать, например, «Договор с клиентом» и другие.

Заключение

Очень хочется, чтобы упоминаемые «лебедь, рак да щука» решили проблему, сместив воз с места, без привлечения кого-то четвёртого, иначе «знаменитое трио» может оказаться не менее

знаменитым «квartetом» из всё той же басни И.А. Крылова.

1. Я.И. Перельман. Занимательная физика, изд. 15-е, Государственное издательство технико-теоретической литературы, М.:1949.
2. Барановский С. Как запустить «САМО-Тур» для Windows под Linux. //Системный администратор, №5, 2008 г. – С. 42-29.
3. Сайт фирмы Etersoft, занимающейся вопросами запуска приложений под Linux – <http://www.etersoft.ru>.
4. Uni Office@Etersoft, транслятор COM-объектов MS Office – <http://wiki.etersoft.ru/UniOffice>.
5. Fast Reports Inc., генератор отчётов, создание и разработка отчётов – <http://fast-report.com/ru>.
6. Сайт программы Dependency Walker, которая ищет зависимости между библиотеками – <http://www.dependencywalker.com>.

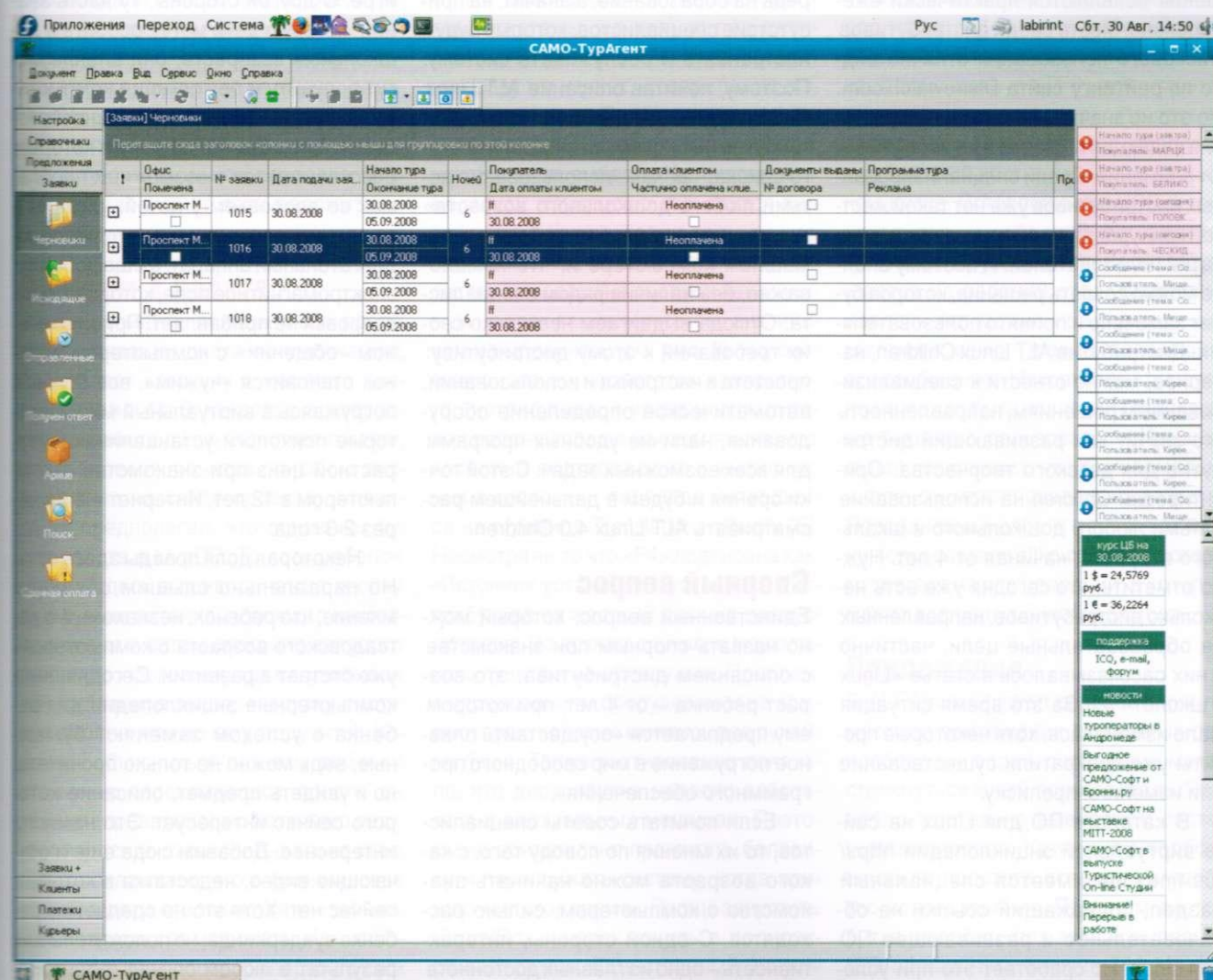
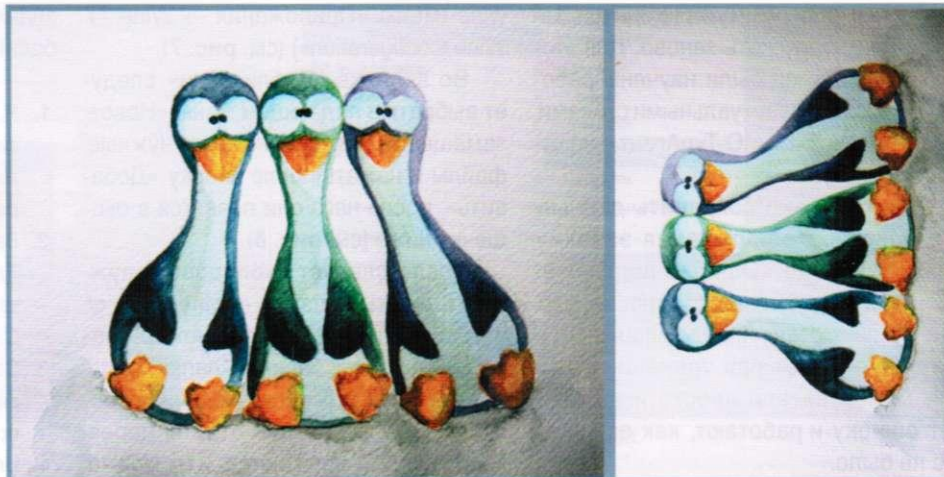


Рисунок 9. Программа «САМО-ТурАгент», работающая под Linux

Обзор ALT Linux 4.0 Children



Сергей Яремчук

В конце июля компания ALT Linux представила бета-версию необычного дистрибутива, предназначенного для детского творчества – ALT Linux 4.0 Children. Познакомимся с новинкой.

Мысли вслух

Дистрибутивов Linux сегодня насчитывается уже не одна сотня, а новые решения появляются практически еженедельно. Места среди дистрибутивов массового пользования отлично видно по рейтингу сайта Distrowatch.com. Но это не значит, что остальные решения являются плохими и не заслуживают внимания. Среди специализированных дистрибутивов уже нет такой жесткой конкуренции и распределения симпатий пользователей. И поэтому вполне можно создать решение, которое будет замечено и принято пользователями. Дистрибутив ALT Linux Children, наверное, можно отнести к специализированным решениям, направленность его звучит как развивающий дистрибутив для детского творчества. Ориентирован Children на использование детьми любого дошкольного и школьного возраста, начиная от 4 лет. Нужно отметить, что сегодня уже есть несколько дистрибутивов, направленных на образовательные цели, частично о них рассказывалось в статье «Linux в школе?» [2]. За это время ситуация мало изменилась, хотя некоторые проекты уже прекратили существование или изменили прописку.

В каталоге ПО для Linux на сайте виртуальной энциклопедии <http://rus-linux.net> имеется специальный раздел, содержащий ссылки на образовательное и развивающее ПО для Linux. Но сработает это при условии, что Linux уже установлен. Все спе-

циализированные дистрибутивы, с которыми удалось за это время познакомиться, ориентированы в первую очередь на образование, а значит, на присутствие специалистов, которые будут настраивать и обслуживать систему. Поэтому, почтав описание ALT Linux Children, делаем вывод, что этот дистрибутив несколько выпадает из этого списка, так как «использование детьми любого дошкольного возраста» подразумевает скорее работу на домашнем компьютере и, что немаловажно, без наличия рядом специалиста. Отсюда выдвигаем несколько своих требований к этому дистрибутиву: простота в настройке и использовании, автоматическое определение оборудования, наличие удобных программ для всех возможных задач. С этой точки зрения и будем в дальнейшем рассматривать ALT Linux 4.0 Children.

Спорный вопрос

Единственный вопрос, который можно назвать спорным при знакомстве с описанием дистрибутива, это возраст ребенка – от 4 лет, при котором ему предлагается «осуществить плавное погружение в мир свободного программного обеспечения».

Если почтавать советы специалистов, то их мнения по поводу того, с какого возраста можно начинать знакомство с компьютером, сильно расходятся. С одной стороны, интерактивность – одно из главных достоинств любой программы – позволяет ребен-

ку общаться с компьютером на равных. Дети быстро осваивают основные манипуляции и с радостью предаются игре. С другой стороны, ту часть знаний, которую дети могут освоить в дошкольном возрасте, они вполне могут получить и без помощи компьютера. Ситуацию обостряют и разнообразные публикации на эту тему, в которых рассказывается о возможных проблемах со здоровьем у детей: ухудшается зрение, плохо развивается опорно-двигательный аппарат, а еще добавим электромагнитное поле, которое также здоровья не прибавляет. При длительном «общении» с компьютером ребенок становится «чужим», все больше погружаясь в виртуальный мир. Некоторые психологи устанавливают возрастную ценз при знакомстве с компьютером в 12 лет, Интернет – еще через 2-3 года.

Некоторая доля правды здесь есть. Но параллельно слышим обратное мнение, что ребенок, незнакомый с детского возраста с компьютером, уже отстает в развитии. Сегодняшние компьютерные энциклопедии для ребенка с успехом заменяют бумажные, ведь можно не только прочитать, но и увидеть предмет, описание которого сейчас интересует. Это намного интереснее. Добавим сюда еще и обучающие видео, недостатка в которых сейчас нет. Хотя это не сделает из ребенка вундеркинда, но положительный результат в любом случае гарантирован. На моем домашнем компьютере

Edubuntu появился на полгода раньше Ubuntu и именно из-за наличия развивающих программ, полезных для моего ребенка. Благодаря ему мы быстро выучили азбуку, математику, узнали про космос, окружающий мир и многое другое. Не говоря уже о том, что просто провели вместе время.

Но есть еще один немаловажный момент. На некоторых форумах мне встречались сообщения о детях, которые в трехлетнем возрасте уже прошли Silent Hill, и родители пытаются для своего чада найти что-то подобное. Естественно, что можно говорить о психике такого ребенка и о подходе родителей? Ведь даже на то, чтобы пройти игру полностью, следует потратить не один час времени. Это очень много. В 12 лет просмотр телевизора не должен составлять больше 30 минут в сутки, а для младшего возраста эти требования еще жестче. Я уже не говорю об игровой атмосфере, которую создали разработчики Silent Hill. Стоит ли говорить, что это явный перебор. Очевидно, взрослым следует понимать, что компьютер не нянька, задача которой отвлечь ребенка, пока родители чем-то заняты. Самостоятельное и бесконтрольное освоение компьютера не может дать положительного результата.

Пробуем в работе

Стильное меню, появившееся после инициализации диска по умолчанию, предлагает загрузку с жесткого диска (см. **рис. 1**). В качестве альтернативы – работа в Live CD и тест памяти. Честно говоря, почта описание дистрибутива на сайте проекта, я почему-то предполагал, что основным будет работа в LiveCD. То есть ребенок вставил диск в привод, нажал кнопку включения питания, подождет некоторое время и затем спокойно запустил любимую программу (даже учитывая все сказанное выше, загрузку системы вполне можно ему поручить). Конечно, объяснить 4-летнему ребенку, что нужно переключиться, просто, он поймет (проверено), но вот на его реакцию я бы сильно рассчитывать не стал. Опыт показывает, что в большинстве случаев дети отвлекаются и «забывают» вовремя среагировать. Как итог – утомительная процедура перезагрузки установленной на жестком диске сис-

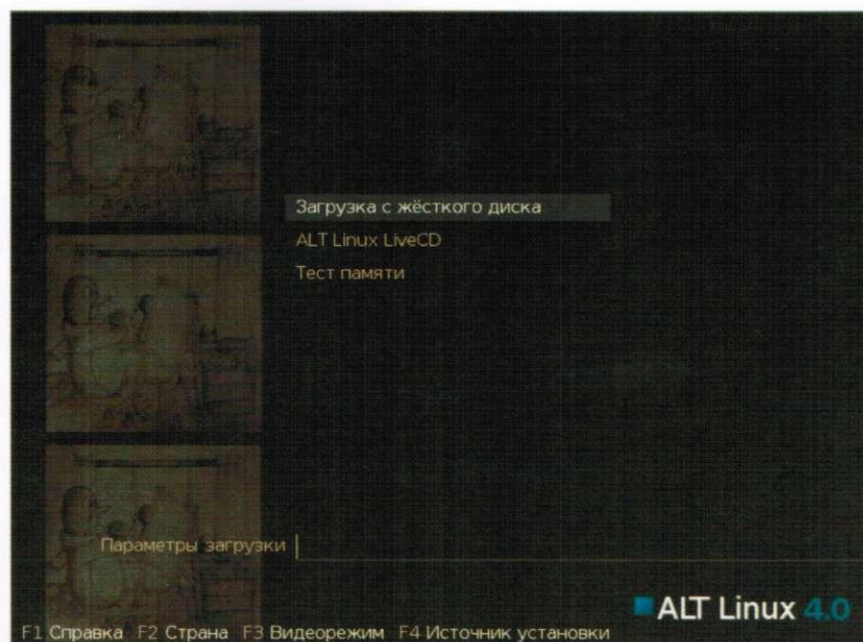


Рисунок 1. Меню загрузки ALT Linux 4.0 Children

темы, терпеть-то ребенок уже не может, ему нужно сейчас и сразу.

По кнопке <F2> можно выбрать язык системы, но только при активации «Россия» и «Украина» система сразу же начнет «говорить» на выбранном языке. Проблем с локализацией в дальнейшем для этих языков нет. Но если выбрать Казахстан или Беларусь, язык меню становится английским, а как показали дальнейшие эксперименты, рабочий стол в таких системах содержит подписи как минимум на трех языках (русском, английском и выбранном) (см. **рис. 2**). Но, очевидно, это не недостаток конкретного дистрибутива, а уровень локализации рабочей среды на этом языке, что косвенно подтверждает и информация о состоянии перевода интерфейса на странице Translations Ubuntu [5]. Несмотря на то что <F4> подписана как «Источник установки», установка дистрибутива на жесткий диск не предусмотрена. Во всяком случае, в меню такого пункта я не обнаружил. Проблем с определением оборудования на компьютерах, которые имелись под рукой, не было. Также тестирование показало, что дистрибутив неплохо работает и в виртуальных машинах. Поэтому не обязательно прожигать болванку и перегружать систему, когда пришло «детское время». Проще запустить VirtualBox и переключить в полноэкранный режим. Единственный минус – низкое разрешение экрана 800x600,

не совсем комфортное на современных мониторах.

Итак, через некоторое время перед нами предстанет рабочий стол KDE 3.5.9. Все разделы жесткого диска, в том числе и NTFS, монтируются в режиме чтение-запись. Их ярлыки выводятся на рабочий стол, обеспечивая быстрый доступ к файлам на жестком диске. Вставленные в USB флеш-накопитель и цифровой фотоаппарат тут же обнаруживаются, ярлык помещается на рабочий стол. Безопасно извлечь его можно при помощи контекстного меню. Правда, появляющееся сообщение «Накопитель отключен, но не может быть извлечен» пугает даже бывалого пользователя. Команда mount показывает, что все нормально и флешку можно удалять из разъема. Повседневная работа ведется от имени пользователя altlinux, прав у которого хватает на все операции, в которых может возникнуть необходимость.

Приложения

В ALT Linux Children разработчики поместили приложения для самых разнообразных задач, с которыми может столкнуться юный пользователь. Этот набор тщательно выверен, и состав минимизирован. Критерием подбора служила простота освоения и удобство использования. При этом дистрибутив не забит «под завязку», размер ISO-образа чуть больше 500 Мб. Меню KDE выполнено традиционно. Все

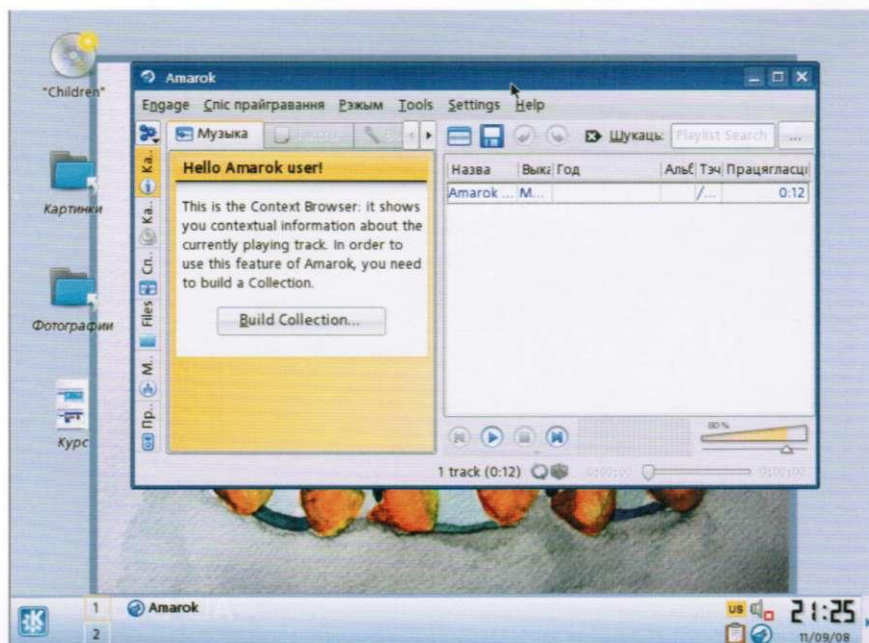


Рисунок 2. При выборе белорусского языка меню будет на 3 языках

пользовательские приложения помещены в четыре пункта меню: «Графика», «Игры», «Интернет» и «Мультимедиа». Плюс в нескольких меню собраны программы для настройки системы, так как в процессе работы сюда заглядывать не понадобилось, то и говорить о них не будем.

Набор приложений для работы с графикой представлен редакторами – растровым (GIMP), векторными (Inkscape и Xara LX) и фрактальным Qosmic, просмотрщиками изображений и работы с фотографиями (showFoto, digiKam и Gwenview). Не забыта и любимая многими детьми веселая рисовалка – TuxPaint. Жаль, что в комплекте всего несколько шаблонов «Штамп». В Edubuntu и некоторых других дистрибутивах этот пункт заполнен несколькими десятками картинок.

В меню «Интернет» найдем только Konqueror и KNetAttach. Здесь нужно отметить, что дистрибутив предназначен больше для развлечения и творчества. Хотя и есть возможность сохранения результатов работы на жесткий диск или флешку, но это не касается настроек системы для последующего восстановления при загрузке. Поэтому Children для повседневной работы не подходит. Убраны все привычные офисные программы, браузеры (Konqueror используется как файловый менеджер), почтовые клиенты, средства планирования, адресные книги

и так далее. Возможности подключения к Интернету урезаны, в графическом меню нет пункта настройки сети или подключения к Интернету. В случае необходимости все действия придется выполнять вручную через etchnet. Наверное, поэтому в списках приложений отсутствуют средства ограничения контента, блокирующие доступ к неблагонадежным ресурсам, вроде DansGuardian, обычно присутствующие в такого рода дистрибутивах.

Меню «Мультимедиа» также не богато приложениями: аудиопроигрыватель Amarok, видеопроигрыватели SMPlayer и Kaffeine, нелинейный редактор Kdenlive.

Кодеки WIN32 в комплекте отсутствуют, поэтому проигрывание видео в некоторых форматах (Real Media и Windows Media 9) невозможно (учитывая специфику, это и не нужно). При щелчке на MP3 файле открывается SMPlayer, который сразу же завис. Не скажу, что это для него характерно, в дальнейшем он без проблем воспроизводит видео. Amarok справился с воспроизведением MP3 без установки дополнительных кодеков.

И наконец, в меню «Игры» можно найти ссылки на 33 игры самых разных жанров – аркады, настольные и карточные игры, простые стратегии (морской бой, минер). Признаться, выбор игр довольно неудачен, действительно образовательные программы практически не представлены. Например, очень

хотелось бы увидеть Gcompris (<http://gcompris.net>), который уже давно стал стандартом де-факто для обучающих дистрибутивов. Нет двух очень полезных программ от проекта Tux4kids [6] – TuxMath и TuxTyping, которые могут помочь в изучении математики и английского языка. Удивляет также полное отсутствие программ из KDE edutainment module [7], хотя проект предлагает 21 программу. И этот список можно еще продолжать.

Из всех представленных программ дети в возрасте 4 и 9 лет могут без проблем освоить TuxPaint, Potato Guy и еще несколько логических игр. Остальные явно рассчитаны на детей старшего возраста.

Кроме этого, на диске есть фотографии и примеры из библиотеки Open Clip Art, ссылки на которые расположены на рабочем столе. Освоить некоторые приемы работы с Digikam, GIMP, Xara LX и Kdenlive поможет краткий обучающий курс.

Минимальная конфигурация компьютера, на которой удалось протестировать ALT Linux Children, – процессор Celeron 633, с 256 Мб ОЗУ, видеокарта Radeon 7000. Проблем и подтормаживания не было. Хотя редактировать большие изображения и видео проблематично, в том числе из-за особенностей работы в LiveCD.

Вот такой он, ALT Linux 4.0 Children, такие дистрибутивы хотя и не занимают топовых позиций в рейтингах, но они нужны и полезны, хотя бы потому, что это лишний повод провести время со своими детьми. Не стоит забывать, что перед нами пока Beta-версия, возможно, в финальном релизе будут изменения, в том числе и в составе программ и утилит.

1. Сайт компании ALT Linux – <http://altlinux.ru>.
2. Яремчук С. Linux в школе? // Системный администратор, №6, 2006 г. – С. 74-76.
3. Сайт проекта Schoolforge – <http://www.schoolforge.net>.
4. Сайт «Linux в образовании» – <http://linuxed.ru>.
5. Страница Translations Ubuntu – <https://translations.launchpad.net/ubuntu>.
6. Сайт проекта Tux4kids – <http://www.geekcomix.com/tux4kids>.
7. Сайт проекта KDE edutainment – <http://edu.kde.org>.

(Продолжение, начало в №7 и №8 за 2008 г.)

Для чего и как производится опечатывание изымаемой техники? Как можно опечатать ноутбук или коммуникатор и какая следственная процедура гарантирует от злоупотребления следствия и от того, чтобы сами следователи могли дописать что-то на системы хранения данных?

Опечатывание изымаемой техники делается для того, чтобы впоследствии доказать неизменность компьютерной информации. Техника опечатывается и упаковывается таким образом, чтобы ее нельзя было достать, не нарушив целостности упаковки. В протоколе осмотра отражается, какой печатью (наименование, номер и принадлежность определенному учреждению печати) опечатана упаковка и чьими подписями данное действие заверено: данные о следователе и понятых. Эксперт в заключении должен указать на неизменность доказательств, а именно: были ли опечатаны разъемы питания и подключения периферии у изъятых компьютеров, не нарушена ли их целостность, если техника упакована – не нарушена ли целостность упаковки.

По правилам ноутбук должен изыматься следующим образом: вытаскивается батарейка, опечатывается разъем под батарейку, разъем под питание, периферию: чтобы не допустить доступ к ноутбуку, либо ноутбук упаковывается таким образом, чтобы его нельзя было достать, не нарушив целостность упаковки.

Коммуникатор и наладонный компьютер (КПК) изымаются в режиме запыпания без выключения, так как после выключения теряется часть информации. При изъятии не допускается нажатия на его сенсорный экран, опечатываются его разъемы и устройство упаковывается так, чтобы доступ к его управлению был невозможен. В противном случае такую улику можно поставить под сомнение.

Во многих сетях применяются системы шифрования. Кроме того, все системы защищаются паролями. Обязаны ли служащие проверяемой компании предоставлять

следствию пароли и коды шифрования?

Такой обязанности не предусмотрено. Всегда можно сослаться на свою забывчивость. Другое дело, что человек на предприятии – это слабое звено: пароли оставляются прямо на рабочем столе, лежат на видном месте. Да и смысла скрывать пароль обычному работнику нет, он не несет ответственности за установленное на его компьютере ПО. Системный администратор и директор могут сослаться на право не свидетельствовать против себя. Даже в случае привлечения в качестве свидетелей это право остаётся и распространяется на информацию, которая может быть в дальнейшем использована против этих лиц.

Подлежат ли досмотру данные, размещенные на миниатюрных устройствах хранения – флешках, карточках, данные на CD? Существует ли в этом случае категория личных вещей, не подвергаемых проверке?

Все зависит от того, с каким постановлением к вам пришли. Если это постановление на обыск, то следователь может потребовать открыть запертые помещения, а в случае отказа вскрыть их; может быть проведен личный обыск – если у следователя возникнут достаточные основания полагать, что лицо, находящееся в помещении, в котором производится обыск, скрывает при себе предметы или документы, которые могут иметь значение для уголовного дела. При проведении осмотра и проверок милиции – досмотр личных вещей и личный обыск возможны только на основании отдельного постановления на выполнение каждого из этих действий.

Кем должен подписываться акт со стороны компании? Если лицо, действующее без доверенности (директор) отсутствует, то кто выпишет доверенность выступать от имени проверяемого юридического лица? Как, в какой форме можно заявить свое несогласие с предметом проверки, с процедурой проверки и с её результатами? Как данное несогласие должно быть зафиксировано проверяющими?

Обзор помещения организации производится в присутствии представите-



Рубрику ведет юрист
Юлия Штокало

ля администрации соответствующей организации. Под представителем организации в данном случае будет пониматься любой управленческий персонал организации, даже не уполномоченный на то руководителем.

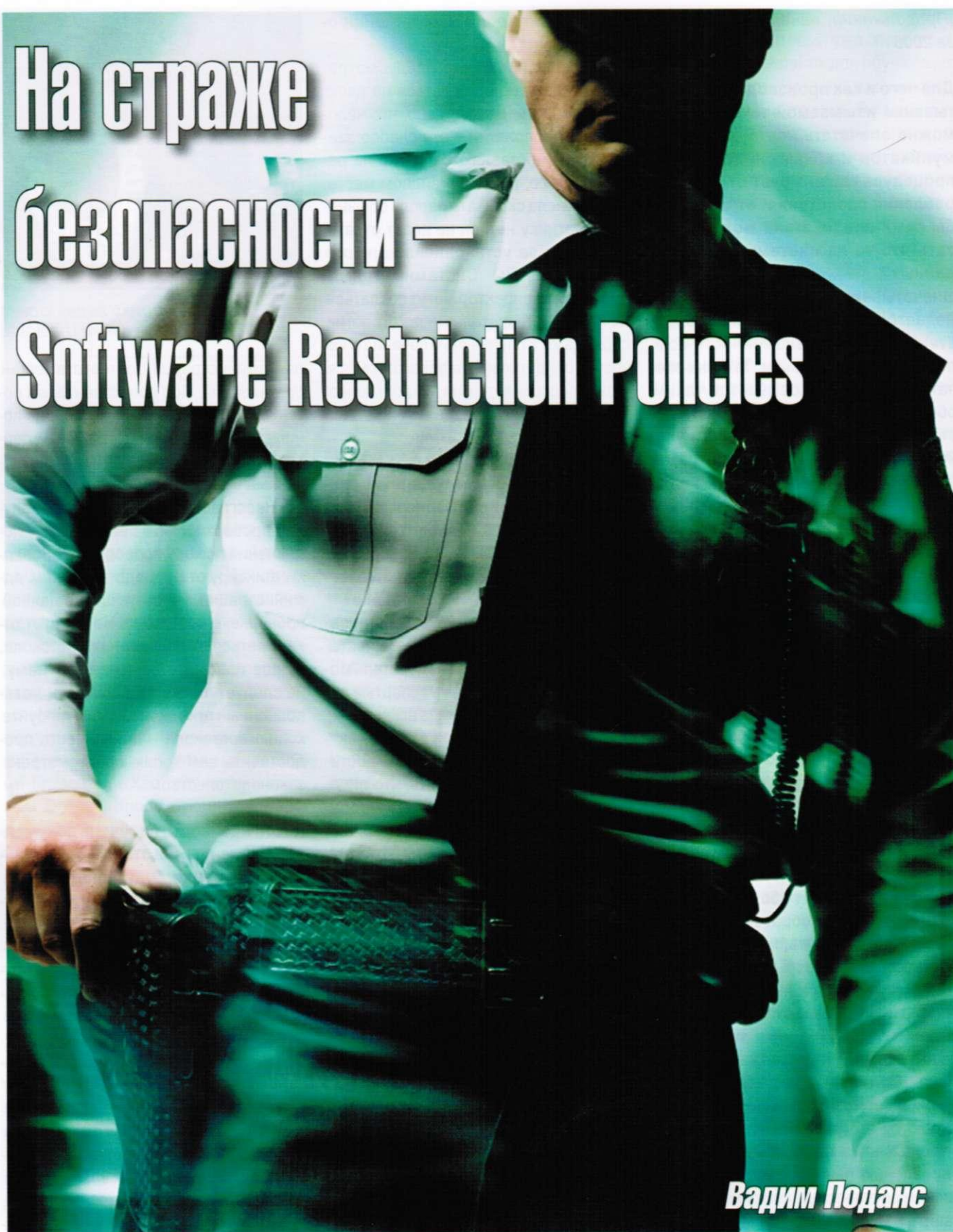
Несогласие с порядком, процедурой проверки, любые замечания и дополнения по ходу проводимой проверки фиксируются представителем администрации, участвующим в данной проверке, в протоколе. Не следует отказываться от подписания протокола, лучше изложите замечания к нему; не следует также подписывать незаполненный протокол; всегда требуйте копию протокола – обязанность предоставить вам копию предусмотрена законодательством.

Какой в точности перечень документов, созданных на этапе и по результатам проверки, будет представлен в суде, например протокол, акты, акты экспертизы и прочее?

Точного перечня документов назвать нельзя. Нет такого положения, которое бы регламентировало, что по одному делу должны быть обязательно представлены на рассмотрение документы согласно определенному списку. Каждый следователь формирует дело исходя из наработанного опыта и знаний. Обязательными являются документы, фиксирующие процессуальные действия, как то: возбуждение уголовного дела, проведение обысков, проверок, привлечение в качестве обвиняемого, назначение производства экспертизы и другие.

Присылайте Ваши вопросы по адресу sekretar@samag.ru или оставляйте на форуме журнала www.samag.ru/forum.

На страже безопасности — Software Restriction Policies



Вадим Поданс

Забитесь о безопасности сети? Для этого необязательно покупать новые дорогостоящие продукты, поскольку зачастую высокоэффективные средства уже вами куплены, но не используются. Научитесь эффективно применять их.

Задачей каждого системного администратора является поддержание в безопасности IT-инфраструктуры своего предприятия, а также ограничивать запуск ненужных пользователю для работы приложений.

Microsoft предлагает интегрированное в линейку Windows Server и линейку корпоративных настольных систем Windows XP/Windows Vista решение, которое называется Software Restriction Policies (сокращённо SRP) – политики ограниченного использования программ.

Данная технология предлагается ещё с линейки Windows 2000 и развивается до сих пор. Тем не менее многие системные администраторы не знают даже о существовании этой политики или избегают её использования из-за сложности настройки. В широком смысле данная политика определяет, какие приложения можно запускать пользователю, а какие запрещено, тем самым снижая риск, что пользователь запустит недозволенное программное обеспечение, скачанное из Интернета, принесённое на флешке или полученное другими противоречащими корпоративной политике способами.

Самый простой способ назначить политику – запустить Group Policy Management Console. Затем перейти в узел: «Computer/User Configuration → Windows Settings → Security Settings → Software Restriction Policies».

При создании политики (кликнуть правой кнопкой мыши и выбрать Create New Policy) она переходит в состояние Unrestricted, то есть не запрещает ничего. Внутри консоли будет 5 объектов (см. рис. 1):

- **Security Levels** – определяет основной уровень безопасности политики по умолчанию.
- **Additional Rules** – здесь задаются исключения для уровня по умолчанию.
- **Enforcement** – окно для выбора степени действия политики.
- **Designated File Types** – окно управления списком расширений файлов, которые проверяются правилом по умолчанию.
- **Trusted Publishers** – представляет настройки управления списками доверенных подписчиков для приложений и скриптов.

Политики SRP имеют 3 уровня безопасности, которые отображены в Security Levels:

- **Disallowed** – запрещено всё, кроме исключений в Additional Rules.
- **Unrestricted** – разрешено всё, кроме исключений в Additional Rules.
- **Basic User** – то же самое, что и Unrestricted, только дополнительно включает запрет запуска приложений с повышенными привилегиями.

Далее следует объект Enforcement (см. рис. 2). В этом окне можно форсировать применение политики не только к исполняемым файлам, но и к связанным с ними библиотекам DLL. Включение данной опции может значительно повысить нагрузку на систему, так как при запуске приложения через политику проходят и все файлы библиотек, из-за чего расходуются и время, и процессорные мощности. Но при этом данная опция позволяет программе при старте подгружать библиотеки только из разрешённых мест. Например, если при запуске программы последняя пытается загрузить библиотеки из профиля пользователя и путь размещения библиотек не указан в исключениях, то политика блокирует их загрузку в память.

Чуть ниже предлагается выбор применения политики для всех пользователей без исключения или для всех пользователей, кроме членов группы Administrators.

Здесь хочу отметить, что при включённом UAC (User Account Control) локальные администраторы не выводятся из-под действия политики SRP и освобождаются от фильтрации только приложения, которые запущены с использованием повышенных привилегий. Также в доменной среде данная опция доступна только в секции Computer Configuration редактора групповых политик.

И последняя опция – Enforce Certificate Rule. На практике правила сертификатов редко используются, и зачастую есть смысл отключить их проверку, что приведет к ускорению обработки политики.

Следом идёт элемент Designated File Type. Здесь перечислен список расширений файлов, которые контро-

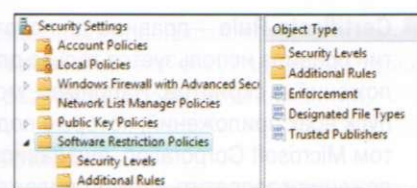


Рисунок 1. Внешний вид консоли управления политикой Software Restriction Policies

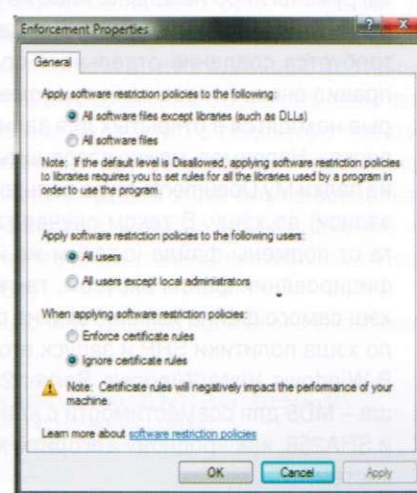


Рисунок 2. Окно выбора степени воздействия политики SRP. Настройки этого окна влияют на глобальную эффективность политик SRP

лируются Software Restriction Policies. Если пользователям необходимо запускать (или запрещено запускать) файлы с указанными в этом списке расширениями, то для них нужно составлять исключения для уровня безопасности по умолчанию (об этом подробнее чуть дальше). Данный список можно редактировать под свои условия.

И последний элемент – Trusted Publishers (см. рис. 3). В этом окне регулируются настройки доверенных издателей сертификатов, которыми подписываются приложения. Данное окно актуально, только если используются правила для сертификатов. В остальных случаях оно не используется. Однако для корректной работы некоторых программ (например, Windows Update в Windows XP/Windows Server 2003) необходимо разрешить управление списком Trusted Publishers для End users (самая верхняя опция).

И, наконец, рабочая область – раздел Additional Rules (см. рис. 4). В этом разделе составляются все исключения для действия политики по умолчанию. Можно использовать следующие типы исключений (дополнительных правил) с учётом порядка их применения:

■ **Certificate Rule** – правило для сертификатов. Данный тип правила используется только для подписанных приложений и скриптов. Например, можно разрешить запуск всех приложений, которые подписаны сертификатом Microsoft Corporation, вне зависимости от их расположения и запретить запуск приложений, которые подписаны Adobe Systems. При этом нужно указать сертификаты соответствующих издателей, которые должны быть загружены либо локально, либо на сетевую папку.

■ **Hash Rule** – правило хэша. Для каждого приложения требуется создание отдельного правила. Данный тип правил очень полезен для приложений и файлов, которые находятся в открытых для записи пользователями папках. Например, можно разрешить запуск приложения из папки My Documents (куда пользователь имеет права записи) по хэшу. В таком случае гарантируется защита от подмены файла (с таким же именем) или при инфицировании файла вирусом, так как в обоих случаях хэш самого файла изменится и не подпадёт под правило хэша политики SRP и запуск его будет невозможен. В Windows Vista/Windows Server 2008 хранится 2 хэша – MD5 для совместимости с клиентами Windows XP и SHA256, как «родной» алгоритм хэширования новых систем.

■ **Network Zone Rule** – правило зоны сети. В Windows реализовано несколько зон сети, как Интернет, Trusted Sites, Restricted Sites, Local Intranet и Local Computer. Данный тип правил регулирует, какие установочные пакеты разрешены для скачивания исходя из условия их размещения. Для корректной работы правил на основе зон сети установочные пакеты должны быть основаны на Windows Installer (иметь расширение .MSI). Например, можно разрешить скачивание пакетов Windows Installer из зоны Trusted Sites и запретить скачивание из зоны Restricted Sites. Однако следует учесть, что данная политика не влияет на скачивание файлов из браузера Internet Explorer. На практике правила зон сети используются достаточно редко.

■ **Path Rule** – правило пути. Правила данного типа позволяют указывать размещение приложений и файлов, которые не будут проверяться политикой и будут разрешены для запуска. По умолчанию уже созданы два разрешающих правила для системных папок Windows и Program Files. Благодаря этому приложения из данных папок будут разрешены для запуска, если, конечно, не удалить правила вручную (чего делать не рекомендуется). Правила для путей могут настраиваться очень гибко, так как позволяют использовать системные и пользовательские переменные окружения (например, %windir%) и знаки подстановок (например, «?» и «*»). Кроме того, можно указать ключ реестра, из которого будет получен путь.

Практическая реализация на примерах

Итак, мы рассмотрели все необходимые опции, которые присутствуют в консоли MMC – Software Restriction Policies. Изучив вышеизложенный материал, можно начинать создавать свои правила политики SRP, но для полноты картины я расскажу о некоторых принципах построения надёжных политик и важных вещах при работе с SRP.

Наиболее эффективным применением политик Software Restriction Policies будет разрешение запуска файлов из папок, куда пользователи не имеют права записи, и запрещение запуска файлов из папок, куда пользователи имеют право записи. В таком случае пользователь гарантированно не сможет запустить файл с рабочего стола (куда он имеет права на запись), но может запустить файл из системной папки Windows (куда обычный пользователь не имеет права записи). Этой схеме наиболее полно отвечает уровень безопасности по умолчанию – Disallowed, которое имеет значение – запретить всё, кроме правил, которые указаны в Additional Rules.

Если применить уровень безопасности по умолчанию Disallowed, то для запуска файлов придётся запускать непосредственно сами файлы в папках, в которых они размещены, так как запуск ярлыков из пользовательского окружения запрещён. Поэтому в Additional Rules необходимо будет создать разрешительные правила, которые позволят запускать ярлыки из Start Menu (это ярлыки, которые отображаются в меню Start и «Start → All Programs»). Для этого в Additional Rules необходимо создать правила для пути, которые будут указывать на:

```
C:\Users\%username%\AppData\Roaming\Microsoft\Windows\Start Menu\*.lnk
C:\Users\%username%\AppData\Roaming\Microsoft\Windows\Start Menu\*.lnk
C:\ProgramData\Microsoft\Windows\Start Menu\*.lnk
C:\ProgramData\Microsoft\Windows\Start Menu\*.lnk
```

Здесь используется системная переменная %username%, вместо которой при работе автоматически будет подставляться имя текущего пользователя. Однако следует учесть, что разрешение запуска ярлыка ещё не значит, что разре-

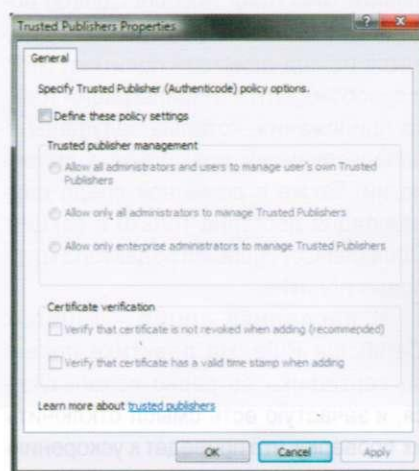


Рисунок 3. Окно с настройкой прав пользователей для редактирования списка доверенных издателей сертификатов и поведения системы при проверке сертификатов

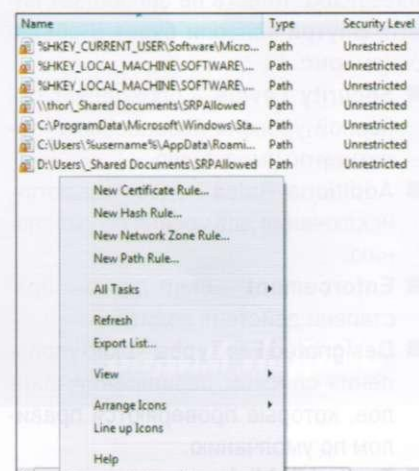


Рисунок 4. Основное окно, в котором администратор создаёт правила исключений. Правила создаются нажатием правой кнопки мыши и выбором нужного типа исключения

шено запускать программу, на которую указывает сам ярлык, еще должно быть разрешающее правило и для самой программы. То есть пользователь не сможет просто изменить путь, на который указывает ярлык для программы в меню «Пуск».

Для этих двух правил необходимо выставить уровень Unrestricted, в результате чего пользователи смогут спокойно запускать ярлыки из меню Start и из вложенных папок первого уровня (например, «Accessories → Paint.Ink»). Однако для систем Windows XP/Windows Server 2003 и систем с локализацией на других языках эти пути могут изменяться в зависимости от языка системы.

Для унификации работы с папками профилей пользователей политика SRP предлагает возможность чтения значений из реестра. Ветка реестра: HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders\ содержит пути для большинства пользовательских папок профиля. Для указания пути к Start Menu рекомендуется использовать значения реестра для каждой локальной машины. Чтобы использовать значения реестра, его ключ нужно заключить в знаки процента «%», как это показано на примерах:

```
%HKEY_CURRENT_USER\Software\Microsoft\Windows\
CurrentVersion\Explorer\Shell Folders\Programs%.lnk
%HKEY_CURRENT_USER\Software\Microsoft\Windows\
CurrentVersion\Explorer\Shell Folders\
Start Menu%.lnk
```

Для того чтобы разрешить запуск ярлыков из меню быстрого запуска (Quick Launch), нужно добавить правило для пути:

```
%userprofile%\AppData\Roaming\Microsoft\
Internet Explorer\Quick Launch%.lnk
```

Таким образом, пользователям уже разрешается запускать ярлыки из меню Start и Programs, при этом абсолютное местоположение меню Start будет определяться каждой машиной индивидуально. Если необходимо разрешить запуск ярлыков или программ из других локаций, то это достигается добавлением правил пути или хэша, как это показано ниже.

Если у вас на диске D: установлено приложение, которое в силу своих особенностей требует разрешения записи в папку с программой для пользователей (увы, но такие приложения до сих пор встречаются, и с ними приходится считаться), то правило пути типа: D:\Programs\SpecProgram*.exe. Или указание имени программы не будет эффективным решением, поскольку пользователю не составит труда переименовать программу и замаскировать под это имя свою любимую игру или зараженный файл. В таких случаях разумнее использовать правила хэша. Если по такому правилу разрешить только необходимые исполняемые (или скриптовые) файлы программы, то при подмене файлов пользователю не удастся запустить ложное приложение, так как хэш нового файла не будет совпадать и файл не подпадёт ни под одно исключение, в следствие чего запуск будет предотвращён уровнем безопасности по умолчанию, то есть Disallowed.

Вот таким комбинированием типов правил в итоге можно получить достаточно гибкую и эффективную политику

ограниченного использования программ, которая позволит контролировать, что пользователи могут запускать только те файлы, которые разрешены политикой компании.

Полезные приёмы

Как правило, при реализованных политиках SRP используется общий уровень безопасности Disallowed, который позволяет достаточно комфортно и безопасно работать в системе, но вносит ограничение на некоторые операции, такие как установка новых приложений. На время процесса инсталляции приложений или других административных операций требуется временно отключать действие политики по умолчанию. Делать это в редакторе политик не очень удобно. Но есть и другое решение – временная деактивизация через реестр. Для этого можно воспользоваться тремя .REG-файлами, которые временно изменяют уровень безопасности (только для Windows Vista/Windows Server 2008):

■ **SRP_Enable** – включает уровень политики в Disallowed:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\
Windows\safer\codeidentifiers]
"DefaultLevel"=dword:00000000
```

■ **SRP_Disable** – включает уровень политики в Unrestricted:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\
Windows\safer\codeidentifiers]
"DefaultLevel"=dword:00040000
```

Самый удобный способ приобретения ПО

Интернет-супермаркет программного обеспечения

открыто 24 часа

Круглосуточный интернет-супермаркет программного обеспечения

www.softkey.ru

Реклама



Рисунок 5. Окно UAC, которое требует подтвердить использование повышенных привилегий или отменить эту операцию

- **SRP_Basic** – включает уровень политики в Basic User:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Safer\CodeIdentifiers]
"DefaultLevel"=dword:00020000
```

И .REG-файлы для Windows XP /Windows Server 2003:

- **SRP_Enable** – включает уровень политики в Disallowed:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\safer\codeidentifiers]
"authenticodeenabled"=dword:00000000
```

- **SRP_Disable** – включает уровень политики в Unrestricted:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\safer\codeidentifiers]
"authenticodeenabled"=dword:00040000
```

Для администраторов рекомендуется использовать логон-скрипт, который будет копировать эти файлы в системную папку и заодно будет копировать ярлыки на рабочий стол администратора при входе на любую рабочую станцию домена. И при необходимости администратор этими ярлыками может временно изменить состояние политики на локальной машине. После перезагрузки системы или очередного обновления групповых политик восстановятся исходные настройки политик.

Что нового в Windows Vista/Server 2008?

Пользователей и администраторов, использующих Windows Vista и/или Windows Server 2008, может заинтересовать уро-

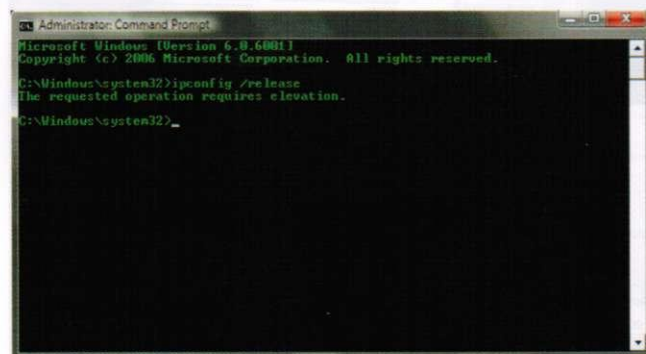


Рисунок 7. Результат выполнения команды ipconfig /release в консоли CMD.EXE с отфильтрованными привилегиями уровнем Basic User

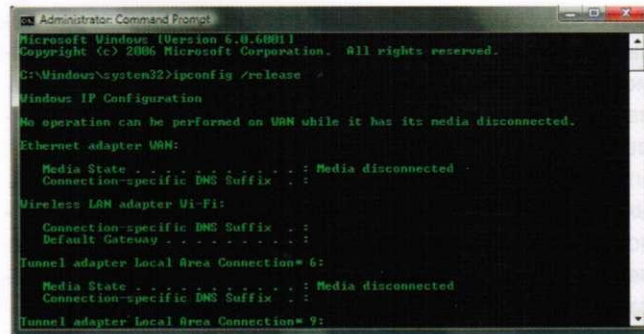


Рисунок 6. Результат выполнения команды ipconfig /release в консоли CMD.EXE с использованием повышенных привилегий

вень безопасности Basic User, который является расширением UAC (User Account Control). В начале статьи ему было дано только определение, а теперь рассмотрим его на практических примерах.

Для начала стоит внести ясность в модель User Account Control. Данный механизм позволяет пользователям с административными учётными записями работать в системе в режиме простого пользователя без использования административных привилегий. То есть локальный администратор в обычном режиме по сути не отличается от обычного пользователя, что обеспечивает защиту от случайных действий администратора.

Когда требуется выполнить административное действие, UAC запрашивает подтверждение на использование административных привилегий, и после подтверждения права администратора значительно повышаются, позволяя изменять настройки системы.

Также UAC позволяет и рядовым пользователям при необходимости повышать свои полномочия до административных путём ввода пароля администратора в окно UAC. Подробнее про UAC можно почитать тут – <http://technet.microsoft.com/ru-ru/magazine/cc138019.aspx>.

Уровень политики SRP – Basic User позволяет пользователям и/или администраторам запускать приложения в обычном режиме (без использования административных привилегий), но запрещает запуск того же приложения с использованием повышенных привилегий.

Для понимания Basic User приведу два наглядных примера.

Пример 1. Запуск консоли CMD для сброса настроек IP всех сетевых интерфейсов

Данное действие из консоли командной строки требует использование повышенных привилегий. Как это выглядит в действительности? Для этого нужно войти в систему под административной учётной записью, кликнуть правой кнопкой мыши на ярлык Command Prompt и выбрать Run As Administrator. После чего UAC попросит подтвердить применение административных привилегий (см. рис. 5).

Подтверждаем эту операцию (нажимаем Continue) и выполняем команду IPCONFIG /RELEASE, которая сбрасывает настройки IP для всех сетевых интерфейсов (см. рис. 6). Команда успешно выполнена, и настройки обнулились.

Для реализации уровня Basic User в Additional Rules создадим хэш-правило для файла CMD.EXE и в качестве Security Level укажем Basic User (см. рис. 8).

Теперь повторяем операцию, подтверждаем использование повышенных привилегий и повторяем команду (см. рис. 7). Как видно из картинки, данная операция не удалась. Хотя мы и запросили повышенные привилегии, уровень Basic User не позволил их получить и запустил консоль в обычном пользовательском режиме.

Пример 2. Запуск System Configuration

Запуск инструментов для диагностирования проблем с запуском системы и автозапуском программ требует использования повышенных привилегий, и без них запустить приложение невозможно. Чтобы продемонстрировать действие Basic User на приложение, которое требует повышенных привилегий, создадим в Additional Rules хэш-правило для этой утилиты, которая находится по адресу: %SystemRoot%\system32\msconfig.exe.

И в Security Level укажем Basic User (см. рис. 9).

Теперь запускаем утилиту из папки Administrative Tools, и снова появляется окно UAC, которое требует подтвердить повышенные привилегии. При подтверждении политика SRP снова по хэш-правилу отменяет эти привилегии и пытается запустить приложение в обычном пользовательском режиме (см. рис. 10).

Однако данное приложение не поддерживает такую работу, в результате чего запуск приложения блокируется полностью действием по умолчанию, как это продемонстрировано на скриншоте.

Таким образом, уровень Basic User позволяет создавать менее жёсткие правила запуска приложений. Например, можно разрешить запуск приложений из любых мест, не опасаясь, что они нанесут ущерб системе, поскольку для деструктивных действий требуются повышенные привилегии.

Заключение

Из вышеизложенного материала следует, что политики ограниченного использования программ – Software Restriction Policies являются гибким и эффективным инструментом в управлении решением контроля запуска приложений пользователями. Однако политики SRP должны быть не един-

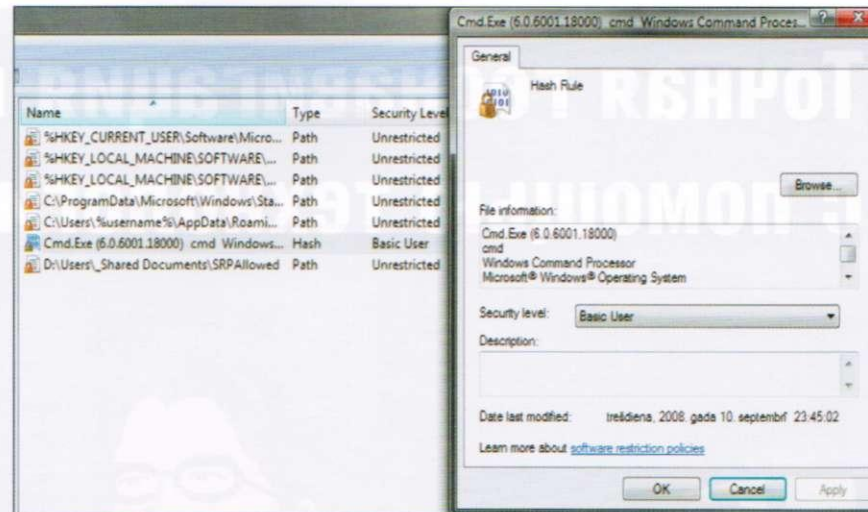


Рисунок 8. Свойства созданного хэш-правила для программы CMD.EXE

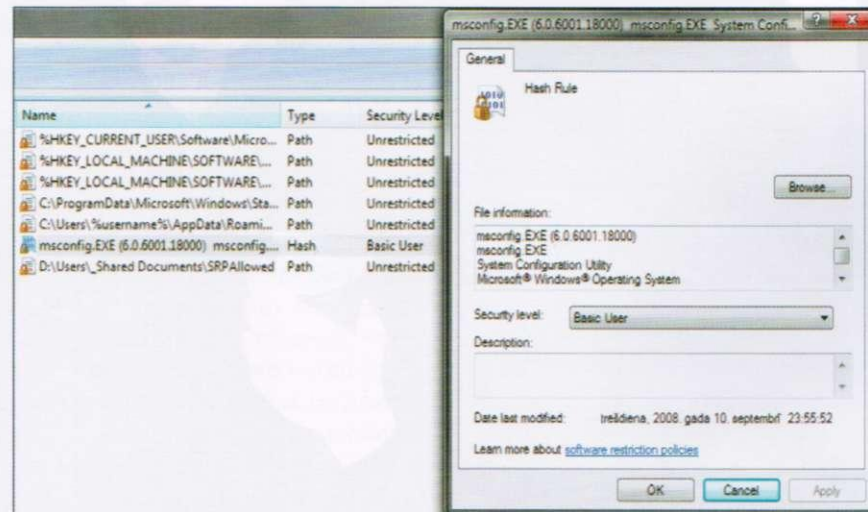


Рисунок 9. Настроенный уровень Basic User хэш-правила для утилиты MSConfig.EXE

ственным средством обеспечения безопасности, а являться частью общего комплекса мер безопасности, и только тогда политики SRP будут по-настоящему эффективны. Фундаментальной частью системы безопасности является работа в системе с минимальными необходимыми привилегиями. Если все пользователи в системе работают под учётной записью с административными привилегиями, то польза от применения политик SRP будет минимальна. Также советуем более ответственно относиться к планированию реализации политик SRP. Чем больше внимания вы уделите стадии планирования, тем меньше времени потребуется для конечной реализации. Кроме того, план поможет избежать ошибок.

1. Software restriction policies overview – <http://technet.microsoft.com/en-us/library/cc759106.aspx>.

2. Using Software Restriction Policies to Protect Against Unauthorized Software (Windows XP/Windows Server 2003) – <http://technet.microsoft.com/en-us/library/bb457006.aspx>.
3. Using Software Restriction Policies to Protect Against Unauthorized Software (Windows Vista/Windows Server 2008) – <http://technet.microsoft.com/en-us/appcompat/aa940985.aspx>.
4. Explore the features: User Account Control – <http://www.microsoft.com/windows/windows-vista/features/user-account-control.aspx>.

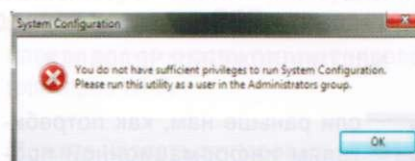


Рисунок 10. Такое сообщение будет получено при попытке запустить системную утилиту без использования повышенных привилегий

Точная геонавигация в городе с помощью технологии Wi2Geo



Антон Борисов

При посещении крупных городов как часто вам приходит в голову мысль: как же легко здесь, наверное, заблудиться? Если частенько, то стоит вас разочаровать. Заблудиться в каменных джунглях становится всё труднее и труднее. Несмотря на отсутствие качественного приема сигналов GPS в местах городской застройки, появляются новые методы определения точного местоположения человека.

Если раньше нам, как потребителям информационной продукции, предлагались решения, с помощью которых мы самостоятельно должны были реализовывать и пре-

доставлять самим себе ту или иную услугу, то сейчас постепенно на главную роль выходит предоставление готовых сервисов и услуг. Удобнее разобрать на конкретном примере. Предположим,

что вы намереваетесь посетить с творческим визитом некий мегаполис. Дома предварительно вы обзавелись картами города для КПК или ноутбука. Естественно, что начертание улиц

и шоссе в нем будет почти в актуальном состоянии. Чего нельзя будет сказать о расположении гостиниц, кафе, магазинов. Понятно, что кто-то постоянно производит мониторинг подобного рода информации и поддерживает карты в должном состоянии. Но карты надо еще найти, а затем, уже будучи на месте, попытаться сориентироваться с помощью того же GPS. Человек – существо своеобразное. Ему хочется если уж не «всего и сразу», то уж точно в виде «здесь и сейчас». То есть хочет знать свое расположение без всякого поиска карт, их закидывания в КПК и уж точно без долгого ожидания, пока приемник GPS выдаст точные координаты. Благо быстро и точно определить свои координаты в городских условиях уже можно.

Как это реализуется на практике, в повседневной жизни? Предлагаю с автором идеи Денисом Алаевым рассмотреть организацию услуги Wi2Geo, которой и посвящено сегодняшнее интервью.

Денис, в чем заключается идея проекта?

Основой проекта является технология позиционирования в городе при помощи Wi-Fi-сетей. Принципиальное отличие от существующих систем навигации (GPS) заключается в интерактивности, т.е. создаваемый сервис будет не только средством для определения местоположения, но и инструментом общения, обмена информацией (текущее местоположение, история перемещений). Среди прочих отличий и преимуществ важно отметить:

- возможность навигации в любой точке города, в том числе на узких улицах и внутри зданий;
- существенно меньшее, чем у GPS, время, требуемое для приведения устройства в рабочий режим;
- более высокая точность по сравнению с GPS и системами GSM-позиционирования (до 1 метра);
- существенно большая аудитория и нулевая стоимость входа (чтобы начать пользоваться, не нужно ничего покупать, достаточно поставить клиентское приложение).

Исходя из вышеперечисленного означает ли, что отправка запросов на серверную часть проекта бу-

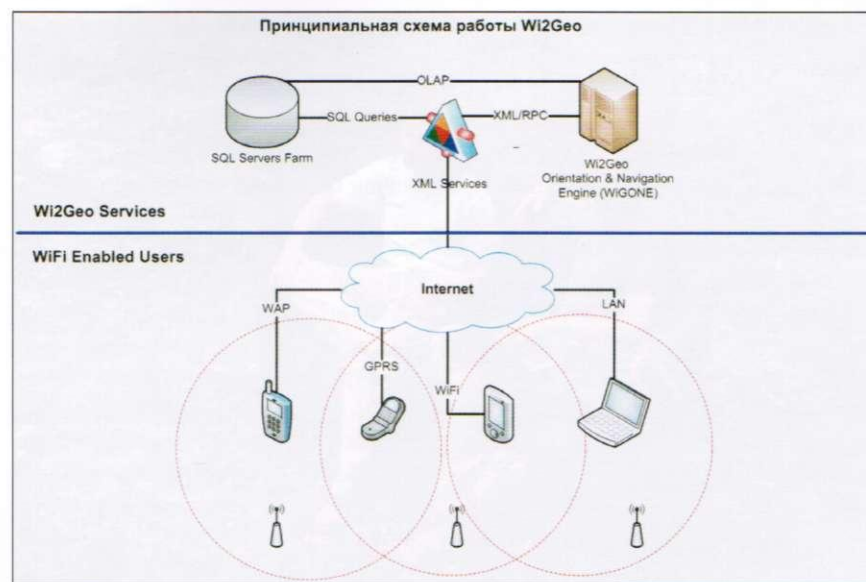


Схема работы Wi2Geo

дет идти через любые интернет-каналы? Предполагается ли клиентскую часть распространять вместе с базой данных точек присутствия Wi-Fi?

Да, отправка запросов на серверную часть будет идти через любые интернет-каналы. Это может быть GPRS, CDMA, тот же Wi-Fi, главное, чтобы клиентское приложение могло соединиться с нашим сервером. На данный момент уже существует приложение для ноутбуков, которое хранит базу данных о Wi-Fi-точках локально, но используется оно только для внутреннего тестирования. Думаю, что в ближайшее время мы начнем распространять базу данных точек вместе с клиентскими приложениями для ноутбуков, это намного увеличит скорость позиционирования и уберет необходимость подключаться к интернет-каналам. Но тем не менее стоимость GPRS-трафика не такая уж и большая, а на каждый запрос приходится не больше 200 байт в обе стороны, так что мы не видим каких-либо серьезных причин добавлять возможность офлайн-позиционирования в мобильные клиенты.

Кого вы рассматриваете в качестве потенциальной аудитории?

Благодаря широкому распространению технологии Wi-Fi и значительному снижению цен на Wi-Fi-совместимые мобильные устройства количество таких устройств и точек доступа существенно выросло в последние го-

ды и продолжает увеличиваться. Согласно исследованиям ABI Research, к 2012 году количество продаж Wi-Fi-чипсетов достигнет миллиарда в год. ABI также прогнозирует, что к 2011 году будет продаваться больше 250 миллионов электронных устройств и более 360 миллионов карманных устройств, оснащенных Wi-Fi. На данный момент почти 100% выпускаемых ноутбуков содержат Wi-Fi-чипсет в базовой комплектации. Все владельцы таких устройств являются потенциальными пользователями нашего сервиса.

Какова бизнес-модель? За счет чего предполагается зарабатывать деньги?

Конечному пользователю основные сервисы будут предоставляться бесплатно. Зарабатывать мы предполагаем на дополнительных сервисах, возможно, на рекламе, а также на продаже доступа к API для сторонних разработчиков, которым интересно добавить возможность Wi-Fi-позиционирования в уже существующие или какие-то новые проекты. То, что такая заинтересованность есть, мы уже уверены – на данный момент у нас есть несколько договоренностей с уже существующими проектами, которые хотят добавить возможность позиционирования через Wi-Fi.

Можете назвать эти проекты или хотя бы их сферу деятельности?

Их огласить пока нельзя, а сфера их деятельности – в основном информа-



Денис Алаев - автор проекта (справа) и Александр Доржиев - один из участников проекта

ционные порталы и социальные сети, есть также пара сервисов, предоставляющих услуги корпоративным клиентам.

А как вы предполагаете вставлять рекламу – что-то наподобие IFRAME или иначе?

Можно и IFRAME, на самом деле это технические детали, посмотрим, вполне возможно, что это будет и JavaScript.

Знакомились ли с зарубежным опытом? Есть ли аналогичные сервисы на рынке?

Да, аналоги есть. Например, на американском рынке действуют две компании близкой направленности: <http://www.skyhookwireless.com> и <http://www.navizon.com>. Первая (по нашим оценкам) достаточно успешна и реализует похожие бизнес-модели в США и Канаде. К описываемому направлению проявляет серьезный интерес компания Intel – они инвестировали в SkyHookWireless и ведут свои научные разработки. На российском рынке в данный момент ничего похожего нет.

Navizon на своей карте покрытия показывает и Москву. Они со своей технологией уже вышли на рынок РФ или это своеобразная промо-акция получилась?

Да, у Navizon есть карта покрытия в Москве и даже Санкт-Петербур-

ге. Но мы не считаем его своим конкурентом. Во-первых, сделать карту покрытия и выйти на рынок – это абсолютно разные вещи. Карту покрытия для Navizon делают пользователи, видимо, так случайно получилось. Мы не думаем, что на данный момент Navizon серьезно смотрит на российский рынок. Во-вторых, карта покрытия в Москве и Санкт-Петербурге уже сейчас у нас больше примерно в полтора раза, чем у Navizon. В-третьих, у нас разные бизнес-модели, Navizon продает свои приложения, что очень сложно, учитывая «российскую» ментальность, мы же будем предоставлять Wi-Fi-позиционирование вместе с ПО бесплатно.

Как именно происходит добавление новых сетей? За счет энтузиастов, целенаправленного обхода или как-то по-другому?

Одновременно и так, и так. Во-первых, мы постоянно объезжаем новые места, увеличивая покрытие. Во-вторых, есть подпроект – <http://labs.wi2geo.ru>, там энтузиасты могут посмотреть статистику по Wi-Fi-точкам, а также залить свои собранные Wi-Fi-точки. Получается очень эффективно, например, в Санкт-Петербурге, Анталии и Иркутске мы никогда не собирали точек, но тем не менее они там есть.

Кроме вас, как идеолога проекта, кто еще занят в проекте?

На данный момент в проекте работа-

ют 4 человека, считая и меня. Мы уже достаточно давно знакомы и работали над несколькими проектами, так что проблем с взаимопониманием нет.

И «как дошли до жизни такой»?

Мы все из Иркутска, переехали в Москву в разное время, но знаем друг друга давно, еще оттуда. Двое из нас окончили матфак ИГУ, еще один – МФТИ, я сам окончил МАИ, сейчас являюсь аспирантом. Когда-то давно вместе работали в одном интернет-сервис-провайдере, потом еще делали несколько интернет-проектов, в частности, своего хостинг-провайдера.

Корни алгоритма и реализации, каковы они?

В рамках моей кандидатской диссертации был изобретен и реализован алгоритм, позволяющий определять местоположение с точностью от 2 до 15 метров. Отлажен механизм сбора информации о точках Wi-Fi-доступа, необходимой для работы сервиса. Было проведено исследование уже зарегистрированных патентов смежной тематики, и мы посчитали, что формулы расчета местоположения – собственно, know-how – позволяют получить нам патент. Поэтому мы подали заявку на его получение.

Дело в том, что сам алгоритм, которым мы пользуемся, – это не триангуляция. Мы составляем карту мощностей распределения сигналов, при этом нам совсем не нужно знать, где по-настоящему находится Wi-Fi-точка, достаточно знать, в каком месте (широта, долгота) и с какой мощностью мы принимали сигнал от данной конкретной точки доступа. На основе этой информации строятся карты распределения мощностей сигналов, и уже по этим картам определяется местоположение мобильного устройства.

Наверняка вы столкнулись со сложностями в реализации, сколько вы открыли для себя технических изюминок?

Технических изюминок оказалось много. Например, мы параллельно делаем навигацию через GSM, выяснилось, что там достаточно сложно сделать триангуляцию. По умолчанию телефон видит только одну соту, но там есть так называемое инженерное ме-

ню, включив которое, можно увидеть больше одной соты. Но доступ к этому меню был только в старых моделях. В современных же телефонах до него практически невозможно достучаться, а значит, и видеть больше одной соты, отсюда не может быть и триангуляции. Приходится определять навигационные координаты только по одной точке, а это, как ни крути, в лучшем случае метров 300-500 (в центре Москвы, там где базовые станции стоят достаточно часто).

Также выяснилось, что несмотря на то что у Google Maps есть API, для Google Maps Mobile такого API нет. Обычное API использовать нельзя, так как урезанный Javascript в телефонах не умеет работать с настоящим API. Пришлось фактически повторять путь Google и писать собственного клиента для мобильных карт. На данный момент мы используем Google Maps Static, правда, с некоторыми доработками, чтобы добавить возможность плавной прокрутки, правильный зум и так далее.

Со сбором информации об окружающих Wi-Fi-сетях проблем особых не было, ни на Windows Mobile, ни на Windows XP/Vista (на момент написания этих строк у нас есть ПО только под эти две платформы). За исключением того, что в ОС Windows Vista лучше пользоваться утилитой netsh, нежели запросами через NDIS, к сожалению, вариант через NDIS работает только в очень ограниченном количестве Wi-Fi-чипсетов, например, с Atheros работает, а с Broadcom – нет.

В апреле вы посещали несколько мероприятий, в том числе и в МАИ. Хотелось бы выяснить ряд моментов: удалось ли найти инвесторов и насколько финансирование из внешних источников актуально для такого стартапа, как ваш?

Да, верно, посещали. В первую очередь из-за того, что я являюсь аспирантом МАИ. А финансирование из внешних источников всегда актуально, причем больше не для того, чтобы сделать технологию, а для продвижения технологии на рынке, особенно это касается нашего стартапа, так как сама по себе идея достаточно нова, и нужно объяснить пользователям, что это удобно и нужно. В данный момент мы ведем

переговоры с несколькими венчурными фондами и компаниями о возможном финансировании проекта. На ранней стадии проект финансировался из средств основателей. Текущие источники финансирования мы не хотели бы называть.

Когда вы знакомились с зарубежным материалом, то каковы текущие применения (или возможные применения) увидели для промышленного сектора, среднего или малого бизнеса? Понятно, что сфера развлечений, по всей видимости, будет лидировать.

Безусловно, есть много применений для бизнеса. Наш сервис можно применять для служб такси, тогда диспетчеры будут точно знать, где находятся их автомобили, также мы видим большую зону применения для курьерских служб и вообще для всех логистических компаний.

То есть привязать можно разные сферы, например, Аэрофлот и туристические агентства? В том плане, что предоставлять местоположение разных структур рядом с аэропортами, местами отдыха.

Конечно, привязать можно абсолютно разные сферы. Навигация также плотно связана с нашей жизнью и всем, что мы делаем, как и телефонная связь. Вот только используемые для навигации средства на сегодняш-

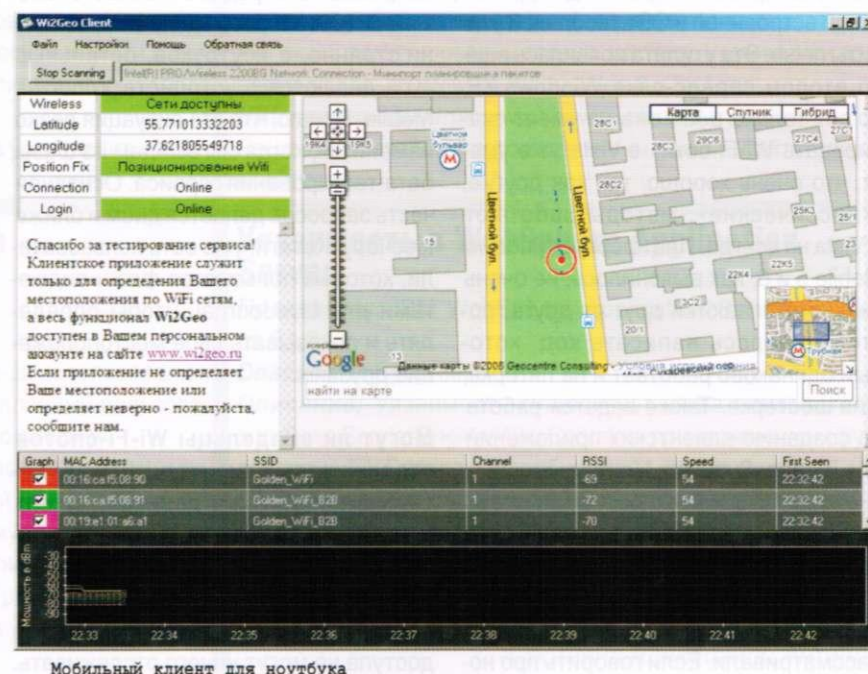
ний день оставляют желать лучшего. Мы пытаемся это изменить.

А выбранный вами инструментарий – операционная система, SQL-движок, механизмы масштабирования, на чем остановили выбор?

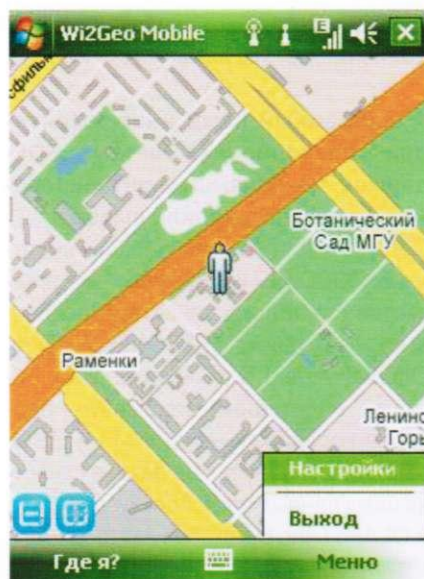
Серверы стоят на операционной системе FreeBSD, мы используем PHP в качестве FrontEnd, в качестве BackEnd выступает движок WiGONE (Wi2Geo Orientation and Navigation Engine), который, собственно, занимается расчетом координат через Wi-Fi и GSM, сам движок написан на Java, в качестве базы данных мы используем MySQL и OLAP4J (the Open Java API for OLAP), в силу специфики алгоритма он нам очень помогает. Технология просто масштабируется, нужно только поставить второй Java-сервер и начать распределять запросы между ними. По нашим расчетам, один такой сервер сможет выдерживать до 100 тысяч пользователей ежедневно.

Почему выбрали FreeBSD? Было ли это сделано исходя из каких-то политических соображений или исходя из прагматических соображений и опыта?

Во-первых, конечно, благодаря опыту. Мы в принципе достаточно давно уже используем FreeBSD (начинали еще с 3.x). Во-вторых, основной движок написан на Java, а, как известно, языку Java – почти все равно, на ка-



Мобильный клиент для ноутбука



Заблудиться можно, но сложно

кой платформе в итоге работает ПО. И к тому же Windows Server нужно покупать, а FreeBSD не просто так имеет приставку Free.

Расскажите, пожалуйста, о клиентской части. Насколько вам удается делать единое ядро программы?

На данный момент, как я уже говорил, клиентское ПО написано для двух платформ Windows Mobile 5/6 и Windows XP/Vista (stand-alone и IE plugin). Для Windows XP/Vista ядро цельное, различаются только участки кода, которые отвечают за сканирование Wi-Fi-сетей. В Windows XP мы сканируем при помощи NDIS, а в Windows Vista сканирование происходит при помощи встроенной в OS консольной утилиты netsh. Эта утилита появилась еще во втором сервис-паке Windows XP, но вот только в XP, она не умеет сканировать Wi-Fi-сети, в Vista уже умеет, что очень хорошо, так как другие, «классические», методы работают в Vista не всегда. Платформы Windows Mobile 5 и 6, как выяснилось, не очень сильно отличаются друг от друга, поэтому удалось написать код, который одинаково работает и на пятерке, и на шестерке. Также ведется работа по созданию клиентских приложений для iPhone, Symbian, Linux.

Рассматривали ли вы возможность сделать унифицированный клиент на Java? Если да, то почему отказались?

Рассматривали. Если говорить про но-

утбуки, то написать унифицированный клиент на Java не представляется возможным, так как даже внутри одного семейства ОС (Windows) приходится прибегать к разным методам опрашивания точек, нечего и говорить, что будет, когда мы попробуем перенести приложение на *nix. В любом случае придется пользоваться JNI, а это не сильно проще, чем написать отдельные приложения для каждой из платформ. Возможно, в будущем мы поменяем политику и сделаем единого Java-клиента, который будет обращаться уже к специализированным демонам на *nix-платформах или системным сервисам на Windows.

С мобильными платформами дело обстоит еще хуже. Из J2ME нет доступа к состоянию Wi-Fi-окружения. Из J2ME даже нельзя получить информацию о GSM-сотах. Есть, конечно, полумифическая спецификация JSR179, по которой можно получить информацию о GSM, но она присутствует буквально на единицах моделей телефонов. Поэтому на мобильных платформах тоже приходится писать native-приложения.

Проводили ли вы аналитические расчеты – какие места больше всего запрашивают и в какое время суток?

По географии у нас пока нет аналитики. Пока можем сказать, что в течение суток делается больше ста тысяч запросов на определение местоположения. 60% запросов делается, как это ни странно, с ноутбуков, остальные 40% делаются с устройств Windows Mobile. Думаю, что эта ситуация резко изменится, когда мы выйдем из фазы бета-тестирования сервиса. Основная часть запросов делается днем и ближе к вечеру. Кстати, уже есть пользователи, которые пользуются флэш-виджетами для LiveJournal, чтобы определять и показывать свое местоположение друзьям.

Могут ли владельцы Wi-Fi-спотов отслеживать клиентов Wi2Geo, которые находятся в данной момент в зоне видимости их сети? Например, для своей внутренней статистики.

По умолчанию владельцы Wi-Fi-точек доступа не могут ничего отслеживать.

Сканирование происходит в пассивном режиме – мы ведь всего лишь настраиваемся на определенную частоту и принимаем сигналы, как радио. Никаких подключений со стороны Wi-Fi-точек не существует. Теоретически, конечно, можно сделать такую возможность со стороны Wi2Geo, т.е. если владелец Wi-Fi-спота докажет, что он владелец, можно говорить ему сколько пользователей (естественно, обезличенных) находится в радиусе действия его сети. Но мне кажется, что в этом нет смысла. Сделать такую статистику не очень просто, а вот толку от нее практически никакого.

Насколько я знаю, у вас запланирована осенняя презентация проекта. Где она будет проходить?

Проходить это будет 18-25 сентября 2008 года в Алуште. Полное точное название: «XVII Международный научно-технический семинар «Современные технологии в задачах управления, автоматизации и обработки информации».

Какие вы видите перспективы до конца года? Что может быть реализовано в рамках города (какие-либо приложения для муниципалитетов, геоинформатики и т. п.)?

До конца года мы планируем значительно расширить покрытие, добавить еще несколько городов-миллионников, написать ПО под максимальное возможное количество платформ (под Windows Mobile 5/6 и Windows XP/Vista уже есть, будет для Symbian, iPhone, Linux, MAC OS), также мы планируем открыть API для возможности писать свои сервисы сторонним разработчикам на основе нашей платформы.

И в качестве послесловия стоит высказать пару своих мыслей. По мере роста городов и более плотной концентрации жителей, такого рода сервисы будут весьма востребованы. Не важно, что конечный пользователь, по всей видимости, не будет обращаться напрямую к серверам Wi2Geo. За него это будут делать социальные сети. А уж в какой форме и что именно они, сети, будут предлагать, покажет будущее. В любом случае, заблудиться скоро будет очень и очень проблематично.

Множественные уязвимости в VMware Server

Программа: VMware Server версии до 1.0.7 build 108231.

Опасность: Высокая.

Описание: 1. Уязвимости существуют из-за неизвестных ошибок в определенных ActiveX-компонентах. Удаленный пользователь может с помощью специально сформированного веб-сайта выполнить произвольный код на целевой системе.

2. Уязвимость существует из-за неизвестной ошибки при обработке запросов в расширении ISAPI. Удаленный пользователь может с помощью специально сформированного запроса вызвать отказ в обслуживании.

3. Уязвимость существует из-за неизвестной ошибки, относящейся к OpenProcess. Локальный пользователь хостовой системы может выполнить произвольный код с повышенными привилегиями на хостовой системе.

4. Уязвимость существует из-за использования уязвимого кода библиотеки freetype. Удаленный пользователь может скомпрометировать целевую систему.

URL производителя: www.vmware.com/products/server.

Решение: Установите последнюю версию 1.0.7 build 108231 с сайта производителя.

Множественные уязвимости в xine-lib

Программа: xine-lib 1.1.14 и 1.1.15, возможно, более ранние версии.

Опасность: Высокая.

Описание: 1. Уязвимость существует из-за целочисленного переполнения в функции `open_ra_file()` в файле `src/demuxers/demux_realaudio.c`. Удаленный пользователь может с помощью специально сформированного RealAudio-файла вызвать переполнение динамической памяти и выполнить произвольный код на целевой системе.

2. Уязвимость существует из-за ошибки проверки границ данных в функции `parse_block_group()` в файле `src/demuxers/demux_matroska.c`. Удаленный пользователь может с помощью специально сформированного Matroska-файла вызвать переполнение динамической памяти и выполнить произвольный код на целевой системе.

URL производителя: xinehq.de.

Решение: В настоящее время способов устранения уязвимости не существует.

Множественные уязвимости в VMware Fusion

Программа: VMware Fusion 1.x.

Опасность: Высокая.

Описание: 1. Уязвимость существует из-за использования уязвимого кода библиотеки freetype. Удаленный пользователь может скомпрометировать целевую систему.

2. Уязвимость существует из-за использования уязвимой версии библиотеки cairo. Удаленный пользователь может скомпрометировать целевую систему. Уязвимости подвержены только VMware Player 2.x for Linux.

URL производителя: www.vmware.com/products/fusion.

Решение: В настоящее время способов устранения уязвимости не существует.

Множественные уязвимости в VMware ACE

Программа: VMware ACE версии до 1.0.7 build 108880 и 2.0.5 build 109488.

Опасность: Высокая.

Описание: 1. Уязвимости существуют из-за неизвестных ошибок в определенных ActiveX-компонентах. Удаленный пользователь может с помощью специально сформированного веб-сайта выполнить произвольный код на целевой системе.

2. Уязвимость существует из-за неизвестной ошибки, относящейся к OpenProcess. Локальный пользователь хостовой системы может выполнить произвольный код с повышенными привилегиями на хостовой системе. Уязвимости подвержены только VMware ACE 1.x for Windows.

URL производителя: www.vmware.com/products/ace.

Решение: Установите последнюю версию с сайта производителя.

Множественные уязвимости в VMware Workstation

Программа: VMware Workstation версии до 5.5.8 build 108000 и 6.0.5 build 109488.

Опасность: Высокая.

Описание: 1. Уязвимости существуют из-за неизвестных ошибок в определенных ActiveX-компонентах. Удаленный пользователь может с помощью специально сформированного веб-сайта выполнить произвольный код на целевой системе.

2. Уязвимость существует из-за неизвестной ошибки, относящейся к OpenProcess. Локальный пользователь хостовой системы может выполнить произвольный код с повышенными привилегиями на хостовой системе.

3. Уязвимость существует из-за использования уязвимого кода библиотеки freetype. Удаленный пользователь может скомпрометировать целевую систему.

4. Уязвимость существует из-за использования уязвимой версии библиотеки cairo. Удаленный пользователь может скомпрометировать целевую систему. Уязвимости подвержена только VMware Workstation 6.x for Linux.

URL производителя: www.vmware.com/download/ws.

Решение: Установите последнюю версию с сайта производителя.

Уязвимость в WebSphere Application Server

Программа: IBM WebSphere Application Server версии до 6.1.0.19.

Опасность: Средняя.

Описание: Уязвимость существует из-за неизвестной ошибки в Servlet Engine/Web Container при включенном функционале FileServing. Подробности не раскрываются.

URL производителя: www-306.ibm.com/software/webservers/appserv/was.

Решение: Установите исправление Fix Pack 19 (6.1.0.19) с сайта производителя.

Составил Александр Антипов

Ричард Столлман: свобода без ограничений



Ричард Столлман – выдающийся мыслитель и деятель, одним из первых поднявший вопрос о соотношении морали и современного законодательства об авторском праве и предложивший эффективное решение в виде лицензий и самого проекта GNU.

Алексей Новодворский,
заместитель генерального директора
компании ALT Linux

Пожалуй, нет в мире человека, знакомого со свободным программным обеспечением и не знающего Ричарда Столлмана – этого добродушного и весёлого человека, подарившего миру идею бесплатного ПО с открытым исходным кодом, яркого противника авторского права. В этом году Столлману исполнилось 55 лет, и он полон энергии для борьбы за свободу человека в ставшем уже таким родным электронном мире.

Ричард Мэтью Столлман (Richard Matthew Stallman) – именно такая запись была сделана в свидетельстве о рождении, полученном Дэниелем Столлманом (Daniel Stallman) и Эллис Липпман (Alice Lippman) в день появления их сына на свет. Произошло это радостное событие 16 марта далёкого 1953 года.

Однако герой сегодняшнего рассказа предпочитает своему имени и фамилии сокращённый вариант. В первом издании книги «Словарь хакера» (Hacker's Dictionary) он написал:

«Ричард Столлман – это моё мирское имя; можете называть меня RMS». Таким образом эти три буквы и вошли в обиход, став его визитной карточкой.

Ещё в студенческие годы, когда увлёкшийся программированием Столлман изучал предметы в Гарвардском университете, он открыл для себя чарующий мир хакерства в изначальной его концепции. Во время своей учёбы в высшем учебном заведении RMS также успел поработать в стенах исследовательского центра IBM в Нью-

Йорке, на кафедре биологии Университета Рокфеллера, а также в недрах лаборатории по исследованию искусственного интеллекта (далее AI Lab) Массачусетского технологического университета (MIT).

Позже, окончив Гарвардский университет и получив степень бакалавра по физике, Столлман перебрался в MIT, где продолжил постигать данную дисциплину. Однако в стенах этого заведения с учёбой у RMS не сложилось, и спустя короткий промежуток времени он оставил планы по получению

учёной степени кандидата физических наук, продолжив работать в AI Lab. Стоит, однако, отметить, что впоследствии Ричард был четырежды удостоен почётной докторской степени.

Во время своей работы в лаборатории искусственного интеллекта Столлман прослыл ярым противником различного рода запретов и ограничений. Когда в 1977 году лаборатория компьютерных наук Массачусетского университета ввела в обиход систему контроля доступа на основе паролей, RMS нашёл способ их расшифровки и разослал каждому пользователю сообщение, в тексте которого содержался пароль, а также предложение изменить его на пустую строку, дабы организовать открытый доступ к компьютерным системам лаборатории. Приблизительно 20% всех пользователей последовали его совету, хотя в конечном итоге «система» восторжествовала. Тем не менее поступок Столлмана не был забыт, и в будущем послужил примером для других борцов за свободу.

За время своей работы в MIT Ричард успел поработать над текстовым редактором TECO, а также операционной системой для Лисп-компьютеров (Lisp Machine) – ЭВМ общего назначения, также разработанных в недрах AI Lab. Кроме того, именно в этой лаборатории RMS создал свой знаменитый текстовый редактор Emacs.

Изначально Emacs представлял собой набор функциональных улучшений и макросов для TECO, написанный Столлманом и Гаем Стили-младшим (Guy L. Steele, Jr.) в 1976 году. Спустя несколько лет, в 1984 году, RMS переписал Emacs на языке программирования C, намереваясь сделать его базовым текстовым редактором для своего нового проекта – GNU. В результате на свет появился GNU Emacs, о котором речь пойдёт дальше.

Свобода – как много в этом слове...

В начале 1980-х годов хакерское сообщество, являвшееся для Столлмана второй семьёй, начало распадаться. Причиной тому стали постоянно усложняющиеся отношения между программистами и крупными компаниями. Появление переносимого ПО – программ, которые после доработки мог-

ли работать на компьютерах различных типов и архитектур, продемонстрировало, что способность пользователя к модификации и распространению программного обеспечения может стать серьёзной помехой в развитии бизнес-стратегий производителей компьютерных систем. Дабы искоренить возможность запуска собственного ПО на ЭВМ конкурентов, компании перестали поставлять исходные коды программ, а также начали ограничивать права на их копирование и распространение путём регистрации авторских прав на свою продукцию.

Именно тогда Столлман осознал необходимость в свободном программном обеспечении с открытым исходным кодом, который мог бы изменять для своих нужд любой желающий. RMS понимал, что подобными шагами компании сильно ограничивают свободу и права пользователей, и смириться с этим не мог.

На протяжении двух лет, с 1982 по 1983 год, Ричард в одиночку боролся с программистами Symbolics (компания-производитель Лисп-машин, «вышедшая из недр» AI Lab), дабы предотвратить получение ими абсолютного контроля над компьютерными системами в лаборатории. К тому времени, однако, Столлман остался последним из своего поколения хакеров в AI Lab, и сопротивляться ему было всё сложнее и сложнее. Во время кульминации противостояния ему было предложено подписать соглашение о неразглашении конфиденциальной информации, а также выполнить ряд других действий, которые он считал предательством по отношению к своим принципам.

«Мне пришлось столкнуться с дилеммой. Я мог либо принять предложение, тем самым признав, что мир меняется, и настала пора измениться и мне, либо же я мог найти другой, более этический путь решения проблемы», – сказал Столлман. «Если бы я согласился, то впоследствии говорил бы: «Я потратил всю свою жизнь на возведение стен, призванных разделять людей». Поэтому Столлман отказался, решив продолжать бороться с «системой».

В сентябре 1983 года RMS начал работать над проектом GNU, название которого является рекурсивным акро-

нимом GNU's Not UNIX, что в переводе означает «GNU – это не UNIX».

Почему именно UNIX? Компания AT&T, владевшая правами на эту операционную систему, до начала 80-х распространяла её совершенно бесплатно. Однако в новом десятилетии компания, осознав ценность UNIX, решила зарабатывать на ней деньги, в результате чего с 1982 года данная ОС перешла в разряд платных систем. Своим акронимом Столлман выражал протест против решения AT&T ограничить свободу пользователей UNIX.

Суть GNU заключалась в создании свободной операционной системы с открытым исходным кодом, которую любой человек мог бы изменять по своему усмотрению. Тем самым пользователь был сам волен решать, каким образом ОС будет работать на его компьютере. «Если люди не будут иметь доступа к исходным текстам программ, с которыми работают, они станут узниками собственного программного обеспечения», – говорит Столлман. «Свобода без ограничений» – вот главный лозунг, с которым шагает по жизни наш герой.

Параллельно с операционной системой RMS начал писать манифест, который был призван объяснить цели проекта, а также популяризировать идеи свободного программного обеспечения. Ради работы над собственным проектом Ричард в 1984 году принял решение уйти из AI Lab.

Столлман утверждает, что писал текст манифеста GNU, основываясь на принципах, по которым он живёт. Его взгляды на жизнь, мир и сообщество – всё это зашифровано между строк документа. «Мой главный принцип заключается в том, чтобы мы помогали друг другу жить вместе лучше», – говорит RMS. «Совершенствование человеческих знаний является частью моего замысла; каждый человек должен иметь к этому свободный доступ и благодаря всеобщему сотрудничеству чувствовать себя частью сообщества. Эти идеи могут быть применены к различным аспектам нашей жизни, но что касается области программного обеспечения, они ведут прямо к свободному ПО».

В своём манифесте Столлман описывал GNU как «полноценную операционную систему, совместимую с UNIX,



Логотип GNU

которую я мог бы раздавать безвозмездно всем, кто захочет и сможет её использовать... Как только GNU будет создана, любой желающий сможет заполучить хорошее программное обеспечение совершенно бесплатно, прямо как воздух».

Официальная публикация манифеста GNU состоялась лишь полтора года спустя после начала работы над ним – в мартовском номере журнала Dr. Dobbs' Journal за 1985 г.

В тексте была описана концепция, согласно которой любая программа может считаться действительно свободной, если она удовлетворяет четырём базовым «свободам»:

- свобода запускать программу для любых целей;
- свобода изучать принцип работы программы и приспособлять её для собственных нужд;
- свобода распространять копии программы, дабы оказать помощь другим;
- свобода совершенствовать программу и публиковать улучшения, дабы они были полезны всему обществу. Это предполагает свободный доступ к исходному коду программы.

Вслед за манифестом Столлман в 1985 году основал Free Software Foundation – некоммерческую организацию, призванную поддерживать разработчиков свободного ПО и предоставлять им легальную инфраструктуру для написания собственных проектов. «Обмен информацией – это фундаментальная основа человеческой дружбы», – говорит Столлман. Он по сей день является президентом Free Software Foundation.

Сразу же после основания FSF Ричард нанёс ещё один «удар»: он популяризировал концепцию «копилефт» (Copyleft), являющуюся полной противоположностью традиционного подхода к авторскому праву. В отличие от него copyleft обеспечивает невозможность запрещать любому человеку право на использование, изменение и распространение произведения. Другим словом: если разработчик выпускает программу в соответствии с данным принципом, никто не в праве ограничивать свободу её дальнейшего распространения либо модификации другими пользователями.

Концепция copyleft легла в основу написанной Столлманом лицензии на свободное программное обеспечение, получившей название «Универсальная общественная лицензия GNU» (GNU General Public License, либо просто GPL). Целью GPL является предоставление пользователю права копировать, модифицировать и распространять программы, а также гарантирование того, что пользователи всех производных программ также получат вышеперечисленные права.

Первоначально Столлман для реализации GNU занялся текстовым редактором, решив приспособить для этих целей Emacs. В результате на свет появился GNU Emacs, первая версия которого увидела свет в марте 1985 года.

За GNU Emacs последовали компилятор GNU C Compiler, который позднее стали называть GNU Compiler Collection, или GCC, а также отладчик GNU Debugger (GDB). Как текстовый редактор, так и средства разработки были выпущены под лицензией GPL.

К 1990 году у команды GNU на руках была собрана почти вся операционная система. Не хватало лишь самого важного компонента – ядра. В том же году стартовала разработка микроядра Hurd, однако было ясно, что этот процесс может затянуться на очень долгое время...

И тут на сцене появился финский парень Линус Торвалдс (Linus Torvalds), решивший интереса и забавы ради написать при помощи средств разработки GNU собственное ядро для операционной системы. В результате в 1991 году на свет появил-

ся Linux, ставший после перехода год спустя на лицензию GPL «последним кирпичиком» для обеспечения целостности и самостоятельности проекта. С тех пор тандем GNU/Linux известен как наиболее широко используемое свободное ПО в мире.

С тех пор прошли годы упорного труда, принёсшие, однако, немалый «урожай». На сегодняшний день сторонниками FSF разработаны тысячи различных приложений и утилит с открытым исходным кодом, а счётчик зарегистрированных пользователей сайта Sourceforge.net, являющегося виртуальным «домом» для разработчиков свободного ПО, давно перевалил за тысячную отметку.

Публичная жизнь

Стоит отметить, что GNU и FSF не являются для Столлмана «дойными корами», поскольку он не получает зарплаты как таковой. Нет у нашего героя и своего дома с машиной. Ведущий скромный образ жизни Ричард считает, что свободный человек не должен зависеть от источника дохода. Он живёт за счёт выступлений на конференциях, а также получения различных премий и наград, таких как премия от благотворительного фонда МакАртуров в 1990 году и награда фонда Такеды в 2001 году.

Перед публикой Столлман начал выступать ещё в начале 90-х гг. прошлого века, пропагандируя свободное ПО. Среди его лекций наиболее часто читаемыми являются «Проект GNU и движение бесплатного программного обеспечения» (The GNU project and the Free Software movement), «Опасности патентов на программное обеспечение» (The Dangers of Software Patents) и «Авторское право и сообщество в эру компьютерных сетей» (Copyright and Community in the age of computer networks).

На протяжении 2006 года, во время проведения консультаций по поводу выхода в свет третьей версии лицензии General Public License, Столлман добавил в свой список ещё одну речь, в которой он рассказывает о предложенных в GPL изменениях.

Среди множества мероприятий, на которых можно было услышать RMS, стоит отметить международную конференцию фонда Викимедиа под

названием Wikimania, а также конференции разработчиков свободного ПО FOSDEM и LinuxTag.

Выступления Столлмана слушали во всех уголках земного шара. Не стала исключением и Россия, первый раз которую он посетил в 1993 году с целью договора о переводе и издании документации проекта GNU. Второй приезд RMS в страну состоялся в марте этого года, во время которого Ричард прочёл две лекции на тему «Свободное программное обеспечение: этика и практика» на факультете ВМК МГУ и МФТИ.

Борьба за терминологию

Столлман очень дотошен, когда речь касается терминологии. «В английском языке у слова «free» два разных значения — «бесплатность» и «свобода», — отмечает он. «В результате люди часто путают свободное ПО с бесплатным. Однако бесплатность программы ещё не гарантирует её «свободного» статуса. Лишь в случае, когда программное обеспечение удовлетворяет четырём «свободам», заложенным в манифест GNU, оно может считаться «свободным, как воздух». Сегодня распространение получил другой термин — ПО с «открытым исходным кодом» (Open Source), однако Столлман не считает его удовлетворяющим концепции «свободного» программного обеспечения, мотивируя своё мнение тем, что данное название скрывает от пользователя саму суть таких программ — свободу: «Бесплатное ПО — политическое движение, программное обеспечение с открытым исходным кодом — модель разработки».

Не согласен RMS и с тем, что общественность в подавляющем большинстве называет операционную систему GNU/Linux просто Linux. Столлман считает, что данный термин необходимо понимать как «ОС, созданная на основе системы GNU и ядра Linux». «Сообществу известно, что Линус Торвальдс написал Linux для развлечения. И они знают, что Линус ничего не говорил о том, что неправильно запрещать людям распространять и изменять программы, которые они используют. Поэтому если общественность считает, что именно с Торвальдса всё началось и что ему надлежит отдавать почести, оно будет следовать его фи-

лософии, что в конечном итоге ослабит нас и наши принципы, за которые мы боремся».

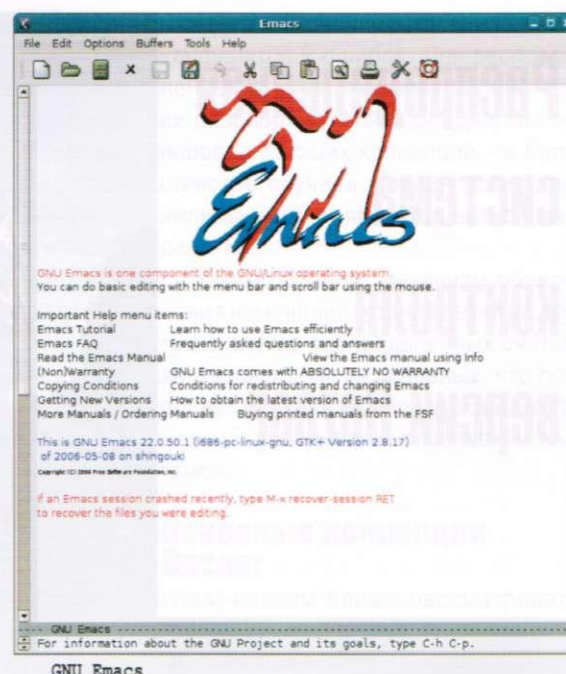
Отношения между Столлманом и Торвальдсом накопились до предела, когда Линус объявил, что существовавшая на тот момент вторая версия лицензии GPL удовлетворяла всем потребностям разработчиков и пользователей, и что не было смысла разрабатывать новую версию: «В мире существует около полусотни различных лицензий для ПО с открытым исходным кодом, и GPLv3 станет лишь одной из них».

На этот счёт у RMS существует своё мнение: «Тот факт, что Линус говорит «ПО с открытым исходным кодом» вместо «свободное программное обеспечение» говорит о его принципах. Я написал GNU GPL для защиты свободы всех пользователей любых версий той или иной программы. Третья версия лицензии призвана лишь ещё лучше выполнять свои обязанности». Столлман считает, что Торвальдс отрицает этот факт, вот почему, возможно, он так не любит GPLv3. «Я уважаю его право выражать свои взгляды, однако считаю их дурацкими. Так что если вы не хотите потерять свою свободу, лучше не идите у него на поводу».

Столлман настолько яростно отстаивает чистоту своей терминологии, что готов общаться с представителями прессы и участниками конференций лишь при условии, что они принимают условия его игры и готовы называть вещи своими истинными именами.

Естественно, среди множества приверженцев и сторонников идей RMS находятся и те, кто в той или иной степени не согласен с ним. Критика в адрес Столлмана звучит за излишне радикальную политику по отношению к любому коммерческому программному обеспечению. Свободолюбивый Ричард непреклонен: «Если вы не можете писать свободное ПО, не создавайте в таком случае вообще ничего».

Нельзя не упомянуть и о спорной дилемме «Что предпочесть — свободу



или же совершенство?» RMS считает: если новая функциональность каким-либо образом влияет на свободу пользователя, её нет необходимости реализовывать.

Однако даже несмотря на подобные разногласия, и критики, и тем более сторонники Столлмана сходятся во мнении, что без влияния, оказанного этим человеком, а также его проектами GNU и GPL на мировую общественность, компьютерная индустрия была бы сегодня совсем другой.

На сегодняшний день Ричард Столлман продолжает отстаивать права и свободы людей. Он протестует против производителей аппаратного обеспечения, пищи и напитков, рекордированных компаний, авторов книг, аэропортов Англии, идентификационных карточек для удостоверения личности... Список акций на его странице stallman.org, которые он организовал либо в которых принимает активное участие, состоит из нескольких десятков пунктов. «Свобода без ограничений», и никаких оговорок!

Дмитрий Мороз

1. <http://www.wikipedia.org>.
2. <http://www.gnu.org>.
3. <http://www.stallman.org>.
4. <http://kerneltrap.org>.
5. <http://cs-exhibitions.uni-klu.ac.at>.
6. <http://superuser.com.au>.
7. <http://www.linuxdevcenter.com>.

Распределенная система контроля версий Vazaa



Дмитрий Васильев

Когда активно работаешь с файлами, будь то исходные тексты программ, файлы конфигурации или статья, то в большинстве случаев нужно использовать какую-либо систему контроля версий. Например, в данный момент я переписываю это введение, и так как я использую систему контроля версий, то смогу в любой момент вернуться к одному из предыдущих вариантов, если мне не понравится этот. Так как основная работа с файлами – это их изменение, то наличие возможности вернуться к предыдущей версии дает огромную уверенность, что, в свою очередь, положительно влияет на производительность. Если работа идет в команде, то система контроля версий необходима, в противном случае в какой-то момент времени работа команды может застопориться.

От централизованных систем к распределенным

Рассмотрим, что сейчас происходит в области систем контроля версий с открытым исходным кодом. Пока это может быть не так очевидно, но просматривается постепенный рост количества распределенных систем и идет яв-

ное движение в эту сторону. Например, из централизованных систем большинство могут назвать следующие:

■ **CVS** (<http://www.nongnu.org/cvs>) – постепенно исчезающая, но все еще популярная система, разработанная поверх формата RCS файлов;

■ **Subversion** (<http://subversion.tigris.org>) – на данный момент наиболее популярная централизованная система контроля версий, изначально разработанная как «лучший CVS» и в итоге исправившая многие ошибки в дизайне CVS;

А что же распределенные системы? Вот наиболее популярные и активно развивающиеся системы:

■ **Git** (<http://git-scm.com>) – распределенная система контроля версий, разработанная Линусом Торвальдсом. Изначально Git предназначалась для использования в процессе разработки ядра Linux, но позже стала использоваться и во многих других проектах – таких, как, например, X.org и Ruby on Rails. На данный момент Git является самой быстрой распределенной системой, использующей самое компактное хранилище ревизий. Но в то же время для пользователей, переходящих, например, с Subversion интерфейс Git может показаться сложным;

■ **Mercurial** (<http://www.selenic.com/mercurial>) – распределенная система, написанная на языке Python с несколькими расширениями на C. Из использующих Mercurial проектов можно назвать такие, как Mozilla и MoinMoin.

■ **Bazaar** (<http://bazaar-vcs.org>) – система, разработка которой поддерживается компанией Canonical – известной своими дистрибутивами Ubuntu и сайтом <https://launchpad.net>. Система в основном написана на языке Python и используется такими проектами, как, например, MySQL и Drupal.

Рассмотрим одну из наиболее гибких, на мой взгляд, распределенных систем контроля версий – Bazaar.

Одним из примеров гибкости Bazaar может служить возможность использования как централизованной модели, так и распределенной и даже смешивание этих моделей контроля версий.

Даже если вы не согласны со мной по вопросу выбора конкретной системы, эта статья поможет вам понять общие принципы работы распределенных систем и затем остановить свой выбор на одной из перечисленных выше.

Зачем нужны распределенные системы?

Как следует из названия, одна из основных идей распределенных систем – это отсутствие четко выделенного цен-

трального хранилища версий – репозитория. В случае распределенных систем набор версий может быть полностью или частично распределен между различными хранилищами, в том числе и удаленными.

Такая модель отлично вписывается в работу распределенных команд, например, распределенной по всему миру команды разработчиков, работающих над одним проектом с открытым исходным кодом. Разработчик такой команды может скачать себе всю информацию по версиям и после этого работать только на локальной машине.

Как только будет достигнут результат одного из этапов работы, изменения могут быть залиты в один из центральных репозиториях или опубликованы для просмотра на сайте разработчика или в почтовой рассылке.

Другие участники проекта, в свою очередь, смогут обновить свою копию хранилища версий новыми изменениями или попробовать опубликованные изменения на отдельной, тестовой ветке разработки.

К сожалению, без хорошей организации проекта отсутствие одного центрального хранилища может быть минусом распределенных систем. Если в случае централизованных систем всегда есть один общий репозиторий, откуда можно получить последнюю версию проекта, то в случае распределенных систем нужно организационно решить, какая из веток проекта будет основной.

Почему распределенная система контроля версий может быть интересна кому-то, кто уже использует централизованную систему – такую как Subversion?

Любая работа подразумевает принятие решений, и в большинстве случаев необходимо пробовать различные варианты: при работе с системами контроля версий для рассмотрения различных вариантов и работы над большими изменениями служат ветки разработки.

И хотя это естественная концепция, пользоваться ею в Subversion непросто. Тем более все усложняется в случае множественных последовательных объединений с одной ветки на другую – в этом случае нужно безошибочно указывать начальные и конечные версии

каждого изменения, чтобы избежать конфликтов и ошибок. Для распределенных систем контроля версий ветки разработки являются одной из основополагающих концепций – в большинстве случаев каждая копия хранилища версий является веткой разработки.

Таким образом, механизм объединения изменений с одной ветки на другую в случае распределенных систем является одним из основных, что позволяет пользователям прикладывать меньше усилий при использовании системы.

Основные концепции Bazaar

Итак, начнем ближе рассматривать Bazaar. На момент написания этой статьи последняя версия Bazaar 1.6.

Но прежде чем рассматривать работу с этой системой контроля версий, сначала обратим внимание на основные концепции, лежащие в ее основе. Если эти термины уже знакомы вам по централизованным системам, они могут иметь немного другое значение в приложении к распределенной системе Bazaar, и их понимание важно для дальнейшего рассмотрения.

■ **Ревизия** – это сохраненное состояние файлов и директорий, включая их содержимое и иерархию в заданный момент времени. С ревизией также связана мета-информация, например:

- ☒ автор изменения;
- ☒ дата изменения;
- ☒ комментарий, связанный с изменением;
- ☒ родительские ревизии, от которых произведена данная ревизия.

После создания, ревизии не меняются и могут быть идентифицированы глобально уникальным номером ревизии (revision-id), например таким: «pqm@pqm.ubuntu.com-20071129184101-u9506rihe4zbzyyz».

Идентификаторы ревизий создаются в момент фиксации (commit) изменений или в момент импорта ревизий из других систем. Идентификаторы ревизий необходимы для функционирования системы, но вряд ли кто-то сможет использовать идентификатор такого типа в разговоре. Специ-

ально для упрощения идентификации ревизий разработаны специфичные для ветки номера ревизий.

Номера ревизий генерируются «на лету» в момент исполнения команд и представляют собой последовательно увеличивающийся номер. Таким образом первая ревизия имеет номер 1, следующая за ней, более поздняя, ревизия – номер 2 и так далее. При объединении изменений с других веток номер ревизии представляется тремя числами, разделенными точками (например: 3112.1.5):

1. Номер ревизии для данной ветки, от которой были произведены объединенные изменения.
2. Порядковый номер (счетчик) объединенной ветки. Так как объединение может затрагивать сразу несколько веток, этот номер служит их счетчиком.
3. Порядковый номер ревизии с момента создания объединяемой ветки.

■ **Рабочее дерево** – это директория под контролем версий, содержащая файлы, которые может редактировать пользователь. Рабочее дерево ассоциировано с веткой. Многие из команд Bazaar используют рабочее дерево для своей работы. Например, команда commit (зафиксировать изменения) создает новую ревизию на основе текущего состояния рабочего дерева.

■ **Ветка** – в простейшем случае это упорядоченная серия ревизий, где последняя ревизия называется верхушкой. Ветки могут разделяться на несколько веток и затем, позже, объединяться обратно, формируя граф ревизий. При этом в рамках ветки может быть основная линия разработки, ревизии в которой помечаются обычными номерами ревизий. Также ветка может иметь другие линии разработки, помечаемые номерами ревизий через точку, как описано выше.

■ **Репозиторий** – это хранилище ревизий. Обычно ветка имеет свой собственный репозиторий, но также несколько веток могут разделять один репозиторий, чтобы уменьшить используемое ветками место на диске.

Такое разделение концепций позволяет более гибко использовать Bazaar, в том числе и как централизованную систему.

Рассмотрим набор сценариев, в основе которых лежат описанные выше концепции:

■ **Разделяемые репозитории (Shared repositories)** – в этом случае рабочее дерево и ветка находятся в одной директории, но репозиторий находится на одну директорию выше, что позволяет хранить информацию о ревизиях только в одном месте. Такой сценарий позволяет не тратить место на полные копии ревизий, если ветки, находящиеся под репозиторием, отличаются незначительно, например, относятся к одному проекту.

■ **Легковесные рабочие копии (Lightweight checkouts)** – ветка и рабочее дерево находятся в разных местах. Этот сценарий похож на использование централизованной системы контроля версий, когда удаленный репозиторий хранит всю информацию о ревизиях, а рабочая копия представляет собой только рабочие файлы и директории.

Мы рассмотрели все основные концепции, лежащие в основе Bazaar, и теперь можно приступить к изучению интерфейса командной строки.

Начинаем работать с Bazaar

Во многих UNIX-совместимых операционных системах Bazaar может быть установлен штатными средствами системы. Для Windows и других ОС, в которых Bazaar недоступен для установки штатными средствами, его можно скачать с официального сайта: <http://bazaar-vcs.org/Download>.

Хотя для Bazaar есть графические интерфейсы, которые также могут быть установлены, и вместе с основной программой мы будем рассматривать только интерфейс командной строки, являющийся основным интерфейсом работы с Bazaar.

Интерфейс командной строки представлен командой bzi, и первое, что можно сделать, это проверить установку следующей командой:

```
bzi version
```

В ответ будет выведена текущая версия Bazaar, пути используемых системой файлов и директорий, а также лицензионная информация (Bazaar использует GPL2). Для получения списка основных команд можно использовать команду help:

```
bzi help
```

После команды help можно указывать любую другую команду, по которой нужно получить подробную информацию, например:

```
bzi help version
```

Или можно вывести все команды Bazaar, в том числе и определенные подключаемыми модулями, командой:

```
bzi help commands
```

Первое, что нужно сделать перед началом работы Bazaar, это указать свое имя и e-mail, которыми будут подписываться все ревизии. Это делается командой whoami, например:

```
bzi whoami "John Doe <john@nowhere>"
```

Эта же команда без параметров выводит текущее установленное имя:

```
bzi whoami
```

```
John Doe <john@nowhere>
```

Если вы хотите использовать разные имена для разных веток, то можно воспользоваться опцией --branch для установки имени только для текущей ветки:

```
bzi whoami --branch "John Doe <john@nowhere>"
```

Конфигурационные файлы Bazaar хранятся в директории \$HOME/.bazaar для UNIX-подобных операционных систем и в директории C:\Documents and Settings\<username>\Application Data\Bazaar\2.0 для Windows. В этой директории хранятся три основных файла конфигурации:

■ **bazaar.conf** – описывает конфигурационные параметры по умолчанию;

- **locations.conf** — описывает конфигурационные параметры для отдельных веток;
- **authentication.conf** — описывает идентификационную информацию для доступа к удаленным серверам.

Каждая ветка также может содержать файл конфигурации, в этом случае он находится в каталоге `.bzz/branch/branch.conf` внутри ветки.

После того, как мы настроили свое имя и e-mail командой `whoami`, эта информация записывается в файл `bazaar.conf`, и в простейшем случае он будет выглядеть так:

```
[DEFAULT]
email = John Doe <john@nowhere>
```

Работаем самостоятельно

Даже если вы работаете в команде, в какой-то момент времени все равно приходится работать одному, и с точки зрения многих команд Bazaar в этом случае мало что меняется.

Если у вас уже есть готовый проект и его надо поставить под контроль версий, нужно начать со следующей цепочки команд:

```
$ cd project-directory
$ bzz init
$ bzz add
$ bzz commit -m "Импортирование проекта"
```

Здесь первая команда `init` создает в рабочей директории `project-directory` служебную директорию `.bzz`, в которой хранятся репозиторий и ветка проекта.

Следующая команда, `add`, рекурсивно добавляет все файлы и директории в рабочем каталоге под контроль версий. Если нет необходимости в рекурсивном добавлении, то можно использовать опцию `--no-recurse` или перечислить необходимые для добавления объекты сразу вслед за командой.

Bazaar может контролировать и, соответственно, добавлять командой `add`, файлы, директории и символические ссылки (для ссылок хранится значение ссылки, а не объект, на который она ссылается).

Если в процессе работы над проектом создаются новые файлы и директории, которые нужно хранить под контролем версий, они также должны быть добавлены командой `add` или внесены в список игнорируемых файлов, описанной ниже, командой `ignore`. Если этого не сделать, файлы будут показываться Bazaar как неизвестные.

И последняя команда, `commit`, фиксирует изменения на ветке и в репозитории, создавая новую ревизию. Опция `-m` позволяет указать комментарий к ревизии прямо из командной строки. Без этой опции для создания комментария будет вызван текстовый редактор.

На данный момент в Bazaar нет жесткого ограничения на размер комментария (единственное ограничение — это достаточно большой, максимальный размер строки в Python), но рекомендуется делать их разумной длины, достаточной для описания сделанных изменений.

По умолчанию команда `commit` фиксирует все изменения на ветке, даже если запущена из-под директории вет-

ки. Если требуется зафиксировать только отдельную директорию или набор файлов, их можно указать вслед за командой, как здесь:

```
bzz commit -m "Изменения" README.txt src
```

Если при создании нового проекта планируется создавать несколько локальных веток, то можно воспользоваться описанным выше разделяемым репозиторием для экономии места:

```
$ bzz init-repo repo
$ cd repo
$ bzz init trunk
$ cd trunk
```

Создание файлов

```
$ bzz add
$ bzz commit -m "Импортирование проекта"
```

Первая команда `init-repo` создает в директории `repo` разделяемый репозиторий, и под директорией `repo` могут храниться различные ветки проекта. Основную ветку мы создаем следующей командой `init`, которая уже была описана выше. В случае создания разделяемого репозитория в директории `repo/.bzz` хранится репозиторий и в директории `repo/trunk/.bzz` ветка.

Если при добавлении мы хотим игнорировать какие-либо объекты, можно воспользоваться командой `ignore`, ее синтаксис:

```
bzz ignore имена объектов
```

Эта команда создаст в корневой директории ветки файл `.bzzignore`, в котором будут записаны имена или шаблоны для игнорирования. Если удобно, то данный файл также можно редактировать в текстовом редакторе.

Глобальные правила игнорирования можно задать в конфигурационном файле `ignore` в каталоге с конфигурацией, `~/.bazaar/ignore` для UNIX-подобных систем.

Кроме задания полных имен и шаблонов, используемых командой оболочкой, таких как `*.py[co]`, `*.o`, можно использовать регулярные выражения:

```
bzz ignore "RE:lib/*\.o"
```

Просмотр изменений

Когда закончен очередной этап изменений, хорошей практикой, прежде чем фиксировать эти изменения в системе контроля версий, может быть их просмотр.

Команда `status` показывает изменения в рабочем дереве с момента последней ревизии:

```
$ bzz status
```

```
modified:
  README.txt
unknown:
  db.tmp
```

Команда `diff` показывает изменения в текстовых файлах в стандартной унифицированной форме:


```
$ bzip diff
```

```
== added file 'README.txt'
--- README.txt 1970-01-01 00:00:00 +0000
+++ README.txt 2008-01-20 14:23:29 +0000
@@ -0,0 +1,1 @@
+файл README
```

Просматривать изменения между заданными ревизиями можно с использованием опции -r:

```
bzip diff -r 100..
bzip diff -r 100..120
```

Первая команда показывает все изменения, начиная с ревизии 100, а вторая – между ревизией 100 и 120.

Для просмотра истории ревизий используется команда log. Bazaar использует иерархическую историю, где объединенные с данной веткой ревизии показываются с отступами в виде дерева. В этом случае визуально проще отделить объединенные изменения от изменений на основной ветке:

```
$ bzip log
```

```
-----
revno: 100
committer: John Doe <john@nowhere>
branch nick: trunk
timestamp: Tue 2008-08-19 16:25:36 +0100
message:
Объединены изменения с работы
-----
revno: 100.1.1
committer: John Doe <john@nowhere>
branch nick: work
timestamp: Tue 2008-08-19 09:54:08 -0500
message:
Добавлен файл README.txt
```

С командой log так же, как и с командой diff, можно использовать опцию -r для указания необходимых ревизий для просмотра.

Для просмотра краткой истории изменений, не включающей описание ревизий с объединенных веток, можно воспользоваться опцией --short:

```
$ bzip log --short
```

```
100 John Doe 2008-08-19 [merge]
Объединены изменения с работы
```

Кроме того, историю ревизий можно просматривать для отдельного файла или директории, добавив его имя после команды:

```
bzip log README.txt
```

Команда cat выводит содержимое файла для ревизии заданной опцией -r:

```
$ bzip cat -r 100 README.txt
```

```
файл README
```

После просмотра изменений их можно зафиксировать командой commit, которая была описана выше.

Выпуск проекта

После некоторого количества ревизий наступает долгожданный момент выпустить очередную версию проекта. Для упаковки новой версии можно воспользоваться командой export, которая упаковывает проект в зависимости от заданного расширения файла. Например, следующая команда создаст ZIP-архив проекта:

```
bzip export ../releases/project-1.0.zip
```

При выпуске каждой версии имеет смысл пометить ее, чтобы в будущем можно было использовать более простые символические имена для работы с этой ревизией. Это можно сделать командой tag:

```
bzip tag version-1.0
```

После этого созданную метку можно использовать в других командах следующим образом:

```
bzip diff -r tag:version-1.0
```

Исправление ошибок

К сожалению, ошибок невозможно избежать, они случаются, и это надо принимать и быть к этому готовым. Bazaar разработан таким образом, что большинство ошибок можно исправлять, в том числе и с использованием отдельных команд. Рассмотрим различные ситуации и их разрешение.

Если вы случайно поставили под контроль версий не то рабочее дерево, то можно удалить служебный каталог .bzip, хотя стоит быть внимательным, чтобы не удалить что-то нужное.

Если случайно под контроль версий был поставлен файл, который нет необходимости хранить под контролем версий, то его можно удалить командой remove, но без использования опций команда не будет удалять измененный файл:

```
bzip add file.txt
bzip remove file.txt
```

```
bzip: ERROR: Can't safely remove modified or unknown files:
added:
file.txt
Use --keep to not delete them, or --force to delete them
regardless.
```

Соответственно, можно использовать опцию --keep, чтобы оставить файл на диске, но удалить его регистрацию или опцию --force, чтобы удалить и регистрацию, и файл. Надо заметить, что если вы удалите файл, не используя Bazaar, то Bazaar будет считать, что необходимо удалить и регистрацию файла.

Команда revert возвращает все файлы и директории в состояние на момент последней фиксации, но при этом сохраняет измененные файлы под другими именами. Она действует рекурсивно, и если нужно вернуть только несколько файлов, то надо быть внимательным и указывать в командной строке только их:


```
bzr revert
```

Если вы случайно сделали фиксацию, то можно использовать команду `uncommit`, чтобы убрать последнюю ревизию и вернуть рабочее дерево в состояние до момента фиксации. Эта команда позволяет также исправлять и комментарии к ревизии:

```
bzr commit -m "Исправлен файл README"
bzr uncommit
bzr commit -m "Исправлен файл README.txt"
```

Кроме того, команда `uncommit` позволяет отменить несколько последних ревизий, например, три последние:

```
bzr uncommit -r -3
```

Если нет необходимости удалять ревизии, а нужно только вернуть рабочее дерево в прежнее состояние, то можно использовать команду `revert` с опцией `-r`:

```
bzr revert -r -3
```

Исправить неправильно поставленные метки можно с помощью команды `tag`, например, перенести метку на другую ревизию можно с помощью опции `--force`, а удалить с помощью опции `--delete`:

```
bzr tag --delete version-1.0
bzr tag --force version-2.0
```

Работаем в команде

После того как мы рассмотрели работу с Bazaar в одиночестве, рассмотрим распределенные возможности при работе в команде (или по такому же сценарию вы можете работать из нескольких мест, например, с работы и из дома). В простейшем случае, если вы работаете в команде, кто-то из разработчиков уже мог создать центральную ветку Bazaar, или при работе в паре это может быть одна из локальных веток другого разработчика, которую вам необходимо скопировать к себе. Кроме работы с файловой системой, Bazaar может работать со следующими протоколами: FTP, HTTP, HTTPS, SFTP и собственным оптимизированным протоколом `bzr`. Одним из простейших способов открыть к ветке доступ для чтения может быть использование HTTP/HTTPS. В этом случае достаточно в конфигурации веб-сервера открыть, с учетом необходимых ограничений, директорию с веткой на доступ, как и любую другую директорию. Для доступа на запись рекомендуется использовать соединение через SSH, которое можно установить, используя схему `bzr+ssh://`. В этом случае автоматически будет использоваться собственный протокол Bazaar через SSH туннель. К сожалению, на данный момент собственный протокол сам по себе не имеет никаких ограничений на доступ.

Для копирования ветки используется команда `branch`, которая по умолчанию копирует все ревизии из указанного места в локальный каталог, создавая новую ветку. В примере ниже мы создаем разделяемый репозиторий для экономики места и затем копируем удаленную ветку, создавая

ветку в каталоге `remote`, и в итоге создаем еще одну копию, копируя локальную ветку в каталог `local`:

```
$ cd projects
$ bzr init-repo project
$ cd project
$ bzr branch bzr+ssh://some.host/some/project remote
```

```
Branched 10 revision(s).
```

```
$ bzr branch remote local
```

```
Branched 10 revision(s).
```

Если теперь в каталоге `remote` мы посмотрим информацию о ветке командой `info`, то увидим ассоциированную родительскую ветку:

```
$ cd remote
$ bzr info

Standalone tree (format: pack-0.92)
Location:
  branch root: .

Related branches:
  parent branch: bzr+ssh://some.host/some/project
```

Хотя возможно большое количество сценариев работы, в нашем примере все локальные изменения будут делаться на ветке в каталоге `local`, а ветка в каталоге `remote` будет отслеживать все изменения, сделанные на удаленной ветке. Но в простейшем случае можно использовать и только одну локальную ветку.

Итак, представим, что мы проделали некоторую работу и создали несколько ревизий на ветке `local`. Теперь мы обновим копию удаленной ветки изменениями, сделанными другими разработчиками, и потом объединим наши изменения. Для обновления ветки нужно выполнить команду `pull` в каталоге `remote`, при этом будут скачаны и добавлены все ревизии, которые были сделаны на родительской ветке после момента последнего обновления:

```
bzr pull
```

Так как изменения на ветке `local` происходили параллельно с изменениями на удаленной ветке, то мы не можем просто сделать `pull`, а должны объединить изменения с помощью команды `merge` в каталоге `remote`:

```
bzr merge ../local
```

Если путь не указан, то `merge` как и `pull` будет использовать путь к родительской ветке. При объединении изменяется только рабочее дерево, чтобы сохранить объединенные изменения, нужно сделать фиксацию командой `commit`. Bazaar использует качественный алгоритм объединения, учитывающий возможные прошлые объединения, и во многих случаях не надо заботиться о таких тонкостях, как начальная и конечная ревизии для объединения.

В случае если в параллельных ветках было изменено одно и то же место в файле, автоматическое объединение может не сработать, и требуется устранить конфликты вручную. При объединении бинарных файлов по умолчанию любые параллельные изменения всегда будут конфликтовать,

но есть возможность подключения модулей, поддерживающих объединение конкретных бинарных форматов. Файлы, в которых не удалось сделать автоматическое объединение, можно посмотреть по команде `status`, или `conflicts`. При обнаружении конфликта Bazaar создаст дополнительно 3 файла для каждого файла с конфликтами:

- **имя.BASE** – файл с содержимым из предыдущей или последней ревизии текущей ветки;
- **имя.THIS** – файл с содержимым из последней ревизии или рабочая копия из текущей ветки;
- **имя.OTHER** – файл с содержимым из другой ветки.

После того, как конфликт был исправлен вручную, нужно использовать команду `resolve`, чтобы сказать Bazaar об этом, при этом ей можно указать только имена исправленных файлов:

```
bzr resolve
```

Также часто бывает полезно определить, кто поменял конкретные строки в файле, для этого в Bazaar можно использовать команду `annotate`:

```
bzr annotate
```

После того как все изменения сделаны на ветке `remote`, нужно их перенести на удаленную ветку, для этого используется команда `push`, которая переносит все недостающие ревизии на родительскую ветку. Если родительская ветка также была изменена, то Bazaar скажет о расхождении веток и нужно будет сначала сделать их объединение и только затем `push`:

```
bzr push
```

Иногда нужно просто передать изменения для просмотра, не обновляя родительскую ветку. В каком-то случае можно опубликовать свою ветку с изменениями через один из протоколов, поддерживаемых Bazaar, или можно послать изменения по e-mail. Для этого в Bazaar есть команды `send` и `bundle`. Команда `bundle` может быть использована для создания пакета с изменениями, который можно приложить к e-mail-сообщению. Команда `send` позволяет послать сообщение с изменениями прямо из командной строки.

Работа в централизованном стиле

Кроме всего прочего Bazaar позволяет работать в централизованном стиле, когда все изменения в момент фиксации сразу идут и на одну центральную ветку. Рекомендуется все центральные ветки располагать в разделяемом репозитории, что может соответствовать корню репозитория в Subversion. Так как в центральном репозитории есть смысл хранить только историю изменений, его можно создать без рабочего дерева. Рассмотрим процесс создания основной центральной ветки, копирование ее на локальную ветку и добавление файлов:

```
$ bzr init-repo --no-trees sftp://central.host/bzr/repo
```

```
$ bzr init sftp://central.host/bzr/repo/trunk
$ bzr checkout sftp://central.host/bzr/repo/trunk
$ cd trunk
$ cp -ar ../some-files/ .
$ bzr add
$ bzr commit -m "Импорт проекта"
```

Здесь мы использовали команду `checkout`, чтобы скопировать и связать удаленную ветку с локальной. Кроме использования этой команды можно связать любые ветки с помощью команды `bind`. В этом случае в момент фиксации все изменения сразу пойдут и на связанную ветку. Если до момента фиксации на связанной ветке произошли изменения, то команда `commit` скажет о них и нужно будет прежде выполнить описанную ниже команду `update`:

```
bzr bind sftp://central.host/bzr/repo/trunk
```

Также в любой момент можно отвязать локальную ветку командой `unbind`, чтобы вернуться к режиму «локальные фиксации и публикация»:

```
bzr unbind
```

Надо заметить, что команда `checkout` копирует всю историю локально. Если в этом нет необходимости, то можно использовать ее с ключом `--lightweight`:

```
bzr checkout --lightweight sftp://central.host/bzr/repo/trunk
```

После того как мы сделали `checkout` обновления из центральной ветки, можно получать командой `update` как и в централизованных системах:

```
bzr update
```

Если при использовании центральной ветки вам в какой-то момент необходимо сделать фиксацию локальной ветки, можно использовать опцию `--local` команды `commit`:

```
bzr commit --local
```

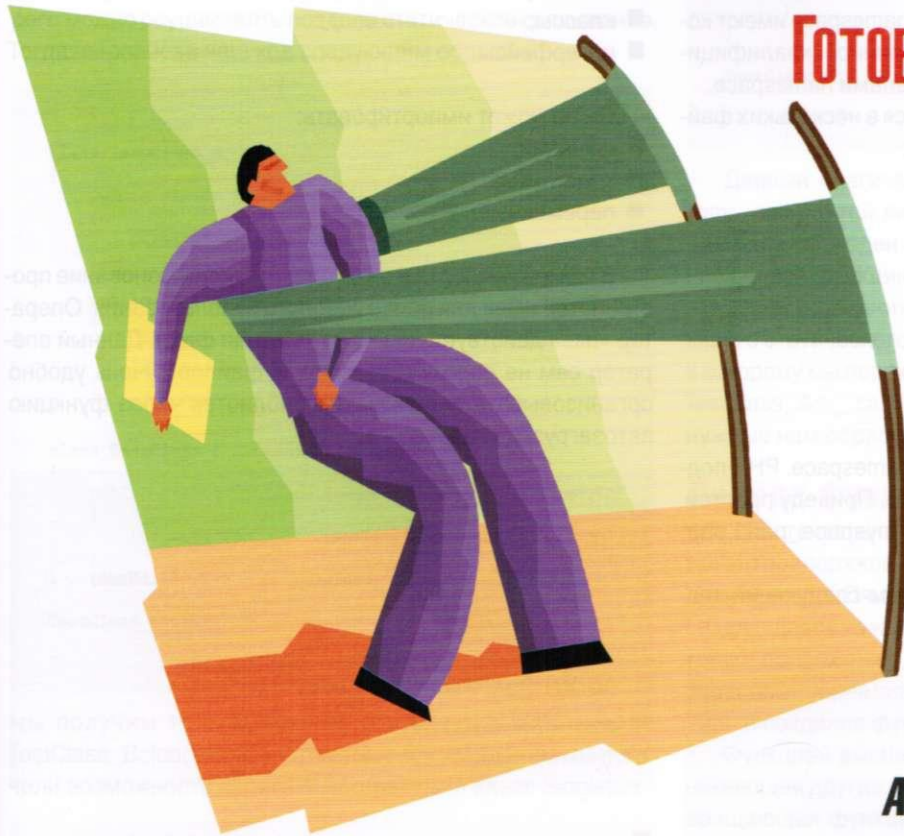
Что дальше?

К сожалению, не удалось включить в эту статью другие возможности или принципы работы Bazaar, о которых хотелось бы рассказать. За кадром остались, например, такие темы:

- подключаемые модули, позволяющие расширять возможности Bazaar;
- как создавать псевдонимы для команд;
- примеры использования Bazaar в различных ситуациях;
- импорт и экспорт ревизий из других систем контроля версий;
- организация веток в виде стека.

Таким образом, эти темы остаются для самостоятельного изучения или могут быть темой одной из будущих статей. В любом случае я уверен, что изучение одной из распределенных систем контроля версий поможет вам увеличить продуктивность вашей работы.

Готовимся к переходу на PHP 5.3



Александр Майоров

Обзор основных нововведений, доступных в PHP 5.3.

РPHP 5.2 существует уже почти 2 года. В нем найдено несколько серьезных ошибок, которые не могут быть исправлены без потери бинарной совместимости. В разрабатываемом PHP 6 из-за перехода на Unicode перестанет работать большое количество наработанного кода, также для него было разработано много интересных дополнений и улучшений, которые не могут быть совместимы со старыми наработками. Для плавного перехода с PHP 5.2 на PHP 6 было решено разработать промежуточную версию PHP 5.3, которая будет содержать большинство улучшений, разработанных для PHP 6, но также способна выполнять существующие скрипты без каких-либо изменений.

Команда разработчиков PHP 1 августа 2008 года с гордостью объявила первый альфа-релиз новой версии PHP 5.3, содержащей значительные изменения по сравнению с предыдущими релизами. Бинарные пакеты для операционной системы Windows будут доступны начиная со 2-й альфа-версии. Промежуточные снапшоты для всех платформ уже сейчас до-

ступны на <http://snaps.php.net>. PHP 5.3, как ожидается, будет отличаться большей стабильностью и производительностью, а также содержать новые функциональные расширения и синтаксические конструкции.

Цель альфа-версии состоит в том, чтобы стимулировать php-программистов не только активно участвовать в выявлении ошибок, но и в обеспечении того, чтобы все новые возможности были протестированы на практике. Также разработчикам важно ваше мнение о новых возможностях языка, которыми вы будете в дальнейшем пользоваться. Вы можете все это сделать через список рассылки либо в багтрекере на php.net.

В этой статье описаны основные нововведения, которые нас ожидают в PHP 5.3. Вот краткий обзор наиболее важных из них:

- пространства имен;
- позднее статическое связывание;
- перегрузка статических методов;
- лямбда-функции и замыкания;
- новое расширение PHAR;
- поддержка SQLite3;
- циклическая сборка мусора;

- поддержка драйвера MySQLnd;
- синтаксические изменения;
- улучшена производительность;
- расширение системы конфигурирования;
- новые возможности SPL.

Пространства имен

Если вы являетесь постоянным читателем нашего журнала, то вы должны иметь представление о пространствах времён из статьи в № 12 за 2007 год. Так что вы можете использовать ту статью как подробную инструкцию по данной теме. Для тех, кто не имел возможности прочесть этот материал, я проведу краткий обзор и опишу дополнительные возможности, которые были добавлены в этой версии.

Пространство имен (namespace) — это область определения переменных, констант и т. п., ограничивающая их область видимости. Оно предназначено для локализации имен идентификаторов и предотвращения конфликтов между ними. Namespace устраняют конфликты имен. Разные namespace могут использовать одно и то же имя для разных целей. Имя внутри про-

странства имеет единственный смысл. Также они делают имена короче. Имена, определенные в namespace имеют короткое (локальное) имя и уникальное длинное (квалифицированное) для использования за пределами namespace.

Один namespace может определяться в нескольких файлах. В namespace могут определяться:

- классы;
- интерфейсы;
- функции;
- константы;
- PHP-код.

В namespace не могут определяться:

- глобальные переменные.

PHP не поддерживает вложенных namespace. PHP поддерживает составные имена namespace. Приведу простой пример работы. Создадим 3 файла: myspace_part1.php, myspace_part2.php, index.php.

В файле myspace_part1.php напомним следующий тестовый код:

```
--- myspace_part2.php ---
<?php
namespace My::Test::Space;

const TEST_ITEM = __NAMESPACE__;

class TestClass
{
    public static function getTestItem()
    {
        return TEST_ITEM;
    }
}

--- myspace_part1.php ---
<?php
namespace My::Test::Space;

require 'myspace_part2.php';

function var_dump()
{
    $args = func_get_args();
    return call_user_func_array('::var_dump', $args);
}

--- index.php ---
<?php
require 'myspace_part1.php';

$Std = new stdClass();
$Std->field_1 = 123;
$Std->field_2 = array(1,2,3);

My::Test::Space::var_dump($Std);

use My::Test::Space::TestClass;
use My::Test::Space as MTS;

$obj = new TestClass;
MTS::var_dump( $obj, MTS::TestClass::getTestItem() );
```

Пример самодостаточен, чтобы на практике показать всю мощь namespace. Еще скажу пару слов про оператор «use», с помощью которого осуществляется импорт. Оператор «use» может импортировать:

- namespace;
- классы;
- интерфейсы.

Он не может импортировать:

- функции;
- константы;
- переменные.

В момент импорта можно сделать переименование пространства имен для более удобного использования. Оператор «use» действует только на текущий файл. Данный оператор сам не подгружает никаких файлов. Очень удобно организовывать подключение библиотек через функцию автозагрузки классов.

```
<?php

function __autoload($libName)
{
    require_once ( str_replace(':', '/', $libName) . '.php' );
}

//require once 'My/Test/Space/testClass.php';
use My::Test::Space::TestClass;

$obj = new TestClass;
```

Позднее статическое связывание (Late Static Binding)

Начиная с версии 5.3.0 PHP реализует возможность, называемую позднее статическое связывание, которая может быть использована для управления статическими методами при наследовании. Для начала приведу небольшой пример кода, на основе которого и разьясню суть проблемы. Есть 2 класса:

```
<?php

class TestClass_A
{
    public static function foo()
    {
        printf("\n %s \n", __CLASS__);
    }

    public static function bar()
    {
        self::foo();
    }
}

class TestClass_B extends TestClass_A
{
    public static function foo()
    {
        printf("\n %s \n", __CLASS__);
    }
}

TestClass_B::bar();
```

В результате выполнения данного кода будет вызван метод TestClass_A::foo(), хотя мы ожидали бы, что будет вызван метод класса TestClass_B. Таким образом, не существовало инструмента для точного определения, какой статический метод должен быть вызван при наследовании. Для решения

данной проблемы был введен класс static, с помощью которого можно осуществить позднее статическое связывание. Тогда, переписав наш код следующим образом:

```
<?php
class TestClass_A
{
    public static function foo()
    {
        printf("\n %s \n", __CLASS__);
    }

    public static function bar()
    {
        // self::foo();
        static::foo();
    }
}

class TestClass_B extends TestClass_A
{
    public static function foo()
    {
        printf("\n %s \n", __CLASS__);
    }
}

TestClass_B::bar();
?>
```

мы получим то, что хотели, а именно вызов метода TestClass_B::foo(). Благодаря этому нововведению мы получили возможность написать абстрактный класс Singleton.

```
<?php
abstract class Singleton
{
    protected function __construct() {}
    protected function __clone() {}

    public function getInstance()
    {
        if ( is_null(static::$Instance) )
            static::$Instance = new static;
        return static::$Instance;
    }
}

class TestSingleton_A extends Singleton
{
    protected static $Instance = NULL;
    public $field = 123;
}

class TestSingleton_B extends Singleton
{
    protected static $Instance = NULL;
    public $field = 456;
}

var_dump( TestSingleton_B::getInstance()->field );
var_dump( TestSingleton_A::getInstance()->field );
?>
```

Перегрузка статических методов

В PHP 5.3 появился «магический» метод __callStatic(), позволяющий перехватывать обращение к статическим методам класса.

```
<?php
class TestClass_A
{
    public static function __callStatic($name, $argv)
    {
```

```
        var_dump( __METHOD__, $name, $argv );
    }
}

TestClass_A::foo('arguments');
?>
```

Данный «магический» метод по сути эквивалентен методу __call(), с той лишь разницей, что он обязательно должен быть объявлен как статический. Уверен, что применение данной функции найдется во многих областях и шаблонах проектирования. Здесь происходит вызов несуществующего статического метода TestClass_A::foo(), обращение к которому мы перехватили с помощью магического метода TestClass_A::__callStatic(). В этом методе мы можем сделать нужные нам обработки и вызвать нужные нам функции.

Лямбда-функции и замыкания

Чем действительно порадовали нас разработчики PHP 5.3, так это поддержкой лямбда-функций, за что им особое большое спасибо! Лямбда-функция – это анонимная функция, создающаяся каждый раз при вызове в определенном контексте приложения. Данный тип функций создан в первую очередь для реализации таких алгоритмов, как «замыкания» и создания функций высшего порядка.

Функцией высшего порядка называется функция, принимающая другие функции в качестве параметров или возвращающая функциональные значения. В PHP была уже возможность создавать динамически анонимные функции с помощью функции create_function(). Но данная функция возвращает имя созданной функции в виде строки. Пример такой приведен ниже:

```
<?php
$lambda = create_function("", 'print __FUNCTION__ . "\n";');
$lambda();

call_user_func($lambda);
?>
```

В примере мы создаем динамическую функцию без имени. Точнее, мы не задаем ей имя, но PHP присвоит ей имя в стиле «lambdaN», где N – порядковый номер анонимной функции. К такой функции мы всегда можем обращаться только через переменную, которой мы присвоили результат выполнения create_function(). Данная функция будет занимать ресурсы точно так же, как и обычная пользовательская функция. Эта запись не является настоящей реализацией лямбда-функции. В PHP 5.3 добавили более менее нормальную поддержку лямбда-выражений, которая выглядит следующим образом:

```
<?php
$lambda = function () {
    { echo 'Test function: ' . __FUNCTION__ . "\n"; };
};

var_dump($lambda);

$lambda();

call_user_func($lambda);
?>
```


Во-первых, обратите внимание на обязательную точку с запятой, без которой будет сгенерирована синтаксическая ошибка. Запись функции теперь является выражением, которое обязательно должно закрываться знаком завершения выражения. Лямбда-выражение является объектом класса Closure. Также константа `__FUNCTION__` внутри контекста лямбда-функций не доступна. Поэтому в нашем примере вызов функции выведет всего лишь надпись «Test function: ». При использовании лямбда-выражений в классах есть небольшие особенности. Например есть класс:

```
<?php
class TestClass_A
{
    public $lambda_foo;

    function __construct()
    {
        $this->lambda_foo = function() { return __CLASS__; };
    }
}

$obj = new TestClass_A();
echo $obj->lambda_foo();

?>
```

Данный код приведет к ошибке, так как метода `TestClass_A::lambda_foo()` не существует. Что же делать? Для таких случаев у объекта типа Closure есть метод `__invoke()`, который вызывает функцию. Поэтому вызов придется осуществлять таким образом:

```
<?php
echo $obj->lambda_foo->__invoke();

?>
```

Может, не особо красиво, но что мешает нам сделать вызов лямбда-функций через перехват обращений с помощью метода `__call()`? Да, собственно, ничто.

```
<?php
class TestClass_A
{
    public $lambda_foo;

    function __construct()
    {
        $this->lambda_foo = function() { return __CLASS__; };
    }

    function __call($name, $argv)
    {
        if ( ! isset($this->$name) )
            && $this->$name instanceof Closure )
            return $this->$name->__invoke($argv);
    }
}

$obj = new TestClass_A();
echo $obj->lambda_foo();

?>
```

Но вообще такое должно встречаться редко. Не стоит увлекаться и использовать лямбда-выражения везде, где попало. Раз мы заговорили о магическом методе `__invoke()`, который был добавлен, то стоит его разобрать

чуть более подробно, но после того как закончим с лямбда-выражениями.

Мы уже достаточно успели поговорить о лямбда-функциях и разобрать несколько примеров, но я до сих пор не показал полный синтаксис записи. Шаблон записи лямбда-выражения выглядит так:

```
function & (parameters) use (lexical vars) { body };
```

Вы обратили внимание на оператор `use`? Да, точно такой же оператор используется для импорта пространств имен. В данном контексте он используется для импорта внешних переменных, относительно лямбда-функции внутрь выражения. Лучше всего показать на примере:

```
<?php
class TestClass_A
{
    function foo_method($x, $y)
    {
        $i = 8;
        $n = 32;

        $closure =
            static function ($x, $y) use ($i, $n)
            {
                return ($x+$i) * ($y+$n);
            };

        return $closure($x, $y);
    }
}

$obj = new TestClass_A();
echo $obj->foo_method(3, 2);

?>
```

В данном случае `$x` и `$y` передаются как аргументы функции, а вот `$i` и `$n` импортируются в нее. Импортированные переменные становятся доступными внутри выражения, как если бы они были частью лямбда-функции. Кстати, вышеприведенный пример реализует такой алгоритм высшего порядка, как замыкание, о которых мы поговорим далее. Также надо сказать про префикс `static`. Вы можете создавать статические лямбда-выражения, точно так же, как статические переменные внутри функций.

Замыкания

Суть замыканий в том, что многие функции можно параметризовать целыми алгоритмами. И очень часто такие алгоритмы можно оформить в виде функций, которые передаются в качестве аргумента. Они, как правило, имеют небольшой размер и необходимы только для маленького участка кода, где они применяются. Отсюда возникает необходимость в синтаксисе, позволяющем описывать такие функции прямо по месту использования, не засоряя пространство имен. Следствием такой необходимости были придуманы лямбда-выражения, о которых мы уже говорили, и такой паттерн функционального программирования, как «замыкания».

Замыкание (англ. closure) – процедура, которая ссылается на свободные переменные в своём лексическом контексте. Замыкание, так же как и экземпляр объекта, есть способ представления функциональности и данных, связанных и упакованных вместе. Также можно сказать, что за-

мыкание – это особый вид функции. Она определена в теле другой функции и создается каждый раз во время её выполнения. В записи это выглядит как функция, находящаяся целиком в теле другой функции. При этом вложенная внутренняя функция содержит ссылки на локальные переменные внешней функции. Каждый раз при выполнении внешней функции происходит создание нового экземпляра внутренней функции с новыми ссылками на переменные внешней функции. Замыкание связывает код функции с её лексическим окружением (местом, в котором она определена в коде). Лексические переменные замыкания отличаются от глобальных переменных тем, что они не занимают глобальное пространство имён. От переменных в объектах они отличаются тем, что привязаны к функциям, а не объектам.

```
<?php
$array = array(1, 2, 3, 4, 5, 6, 7, 8, 9);

$array = array_map( function($element) {
    return $element < 5 ? $element /10 :
        $element *10; }, $array );

var_dump($array);
?>
```

Согласитесь, что такая запись намного удобней, чем написание отдельной функции, особенно если учесть, что данный участок кода должен выполняться всего один раз на протяжении работы всей программы. Ради такого случая засорять программу реализацией отдельной функции нет смысла. К тому же после отработки данного кода ресурсы не будут засорены, так как никакой функции не будет висеть в памяти.

Магический метод __invoke()

Данный метод позволяет работать с объектом как с функцией. Сразу приведу пример:

```
<?php
class TestClass_A
{
    function __invoke($i)
    {
        return $i * 16;
    }
}

$obj = new TestClass_A();
echo $obj(8);
?>
```

Теперь в арсенале программистов ООП-щиков появился еще один новый инструмент для реализации новых интересных алгоритмов. Появилась реальная возможность создавать функции-объекты, а это расширяет объектно-ориентированные возможности языка.

Поддержка драйвера MySQLnd

В PHP5.3 добавлен ряд расширений, одним из которых является MySQLnd. Хотя данное новшество не является PHP-расширением в полном смысле этого слова, так как не предоставляет каких-то новых функций, доступных для PHP-программиста. Оно является всего лишь заме-

ной библиотеки libmysql, оптимизированной специально для PHP. MySQLnd использует менеджер памяти Zend Engine, таким образом не требуется копирование данных из библиотеки libmysql в PHP. Оно использует значение директивы memory_limit, что позволяет контролировать выделяемую память. MySQLnd использует потоки PHP, что делает более быстрыми постоянные соединения. Также данное расширение содержит серию оптимизаций для более быстрого получения результатов. Кроме того, MySQLnd позволяет собирать статистику, что может пригодиться в дальнейшем для оптимизации ваших запросов. MySQLnd используется как для работы mysql, так и для mysqli-библиотек.

Вопрос о том, какая библиотека будет использоваться libmysql или mysqlnd, решается на стадии компиляции PHP. Многими любимым PDO на данный момент не поддерживает работу через MySQLnd. Как заявил Дмитрий Стогов на седьмой международной конференции PHPCONF 2008, поддержку PDO реализовывать не будут, но не исключают, что могут найтись энтузиасты, которые смогут осуществить это и написать расширение. Может быть, этим человеком станете вы, уважаемый читатель.

Новое расширение Phar

В PHP 5.3 появилось новое расширение, по принципу аналогичное концепции JAR-технологии Java. Phar-архив прежде всего придуман для приложений-инсталляторов, а также для удобного распространения целого PHP-приложения или библиотеки в одном файле. В отличие от JAR-архивов, Phar-архивы не нуждаются ни в специальной среде исполнения, ни во внешних средствах для запуска процесса или PHP. Phar-архив используется так же, как и обычные PHP-файлы. Например, чтобы вывести в браузер изображение из архива, достаточно написать:

```
<?php
header('Content-type: image/jpeg');
echo file_get_contents('phar:///path/to/my/test.phar/
resources/image.jpg');
?>
```

Как видите, для подключения используется специальный стрим-враппер 'phar:'. Давайте теперь попробуем на практике создать Phar-архив. Вообще надо сказать, что специальных архиваторов нет. Но это не проблема. Создать такой архив несложно самостоятельно, но есть одно но. По умолчанию работа с Phar-архивами разрешена только на чтение. А чтобы создать архив, нужно разрешить запись. Для этого надо в php.ini добавить всего одну строчку:

```
[phar]
phar.readonly=0
```

Если этого не сделать, ваш скрипт будет все время генерить исключения. Теперь напомним небольшой код:

```
<?php
$DIR = __DIR__;

try {
    $Phar = new Phar('test.phar', 0);

    $Phar['test.php'] =
    <<<'NOWDOC'
```



```
<?php
echo 'This is test phar archive.';

NOWDOC;

$fp = fopen('img.jpg', 'rb');
$Phar['img.jpg'] = fread($fp, filesize('img.jpg'));
fclose($fp);

} catch (Exception $e) {
    echo $e->getMessage();
}

?>
```

Данный код демонстрирует, как в архив добавляется PHP-код из строки и один графический файл. После того как отработает скрипт, будет создан файл test.phar. Проверим, что внутри, делаем простой тест:

```
<?php
$DIR = __DIR__;
include "phar://$DIR/test.phar/test.php";

?>
```

и

```
<?php
$DIR = __DIR__;
header('Content-type: image/jpeg');
echo file_get_contents("phar://$DIR/test.phar/img.jpg");

?>
```

Как видите, ничего сложного. Мы создали исполняемый phar-архив. Данный архив изначально сжат по алгоритму tar. Вы всегда можете сжать данные архивы с помощью ZIP. Добавьте в код следующую запись:

```
<?php
$Phar = $Phar->convertToExecutable(Phar::TAR, Phar::GZ);

?>
```

Если создать в архиве файл с именем index.php, то он будет запускаться автоматически. Достаточно, например, в консоли написать

```
php test.phar
```

либо сделать подключение файла через include:

```
<?php
$DIR = __DIR__;
include "phar://$DIR/test.phar";

?>
```

Более подробную информацию вы можете получить по адресу: <http://ru2.php.net/manual/ru/book.phar.php>

Синтаксические изменения

В самом синтаксисе языка также есть некоторые изменения. Все эти изменения можно отнести к термину «синтак-

сический сахар», но, тем не менее, они приятны. Ведь все мы любим сладкое. Итак, что нам на десерт приготовили разработчики PHP? Давайте посмотрим. Первое – это новый синтаксис старого оператора HEREDOC, который называется NOWDOC.

NOWDOC

В PHP существует синтаксис HEREDOC, который называется «встроенный документ». Обычная форма записи такого оператора выглядит так:

```
<?php
$var_1 = 123;
$var_2 = 456;

echo <<<HEREDOC

Example:
$var_1+$var_2

HEREDOC;

// Example:
// prints 123+456

?>
```

Новый синтаксис NOWDOC показан в следующем примере:

```
<?php
echo <<<'NOWDOC'

Example:
$var_1+$var_2

NOWDOC;

// Example:
// $var_1+$var_2

echo <<<"NOWDOC"

Example:
$var_1+$var_2

NOWDOC;

// Example:
// printd 123+456

?>
```

Суть нового синтаксиса в том, что теперь можно контролировать поведение многострочных вставок. Если раньше они интерпретировались так же, как двойные кавычки, то теперь можно переключить поведение, чтобы данный блок интерпретировался как одинарные. Делается это добавлением одинарных кавычек к открывающему маркеру блока: <<<'NOWDOC'. Без указания кавычек данный блок будет вести себя так же, как и раньше. Для совместимости была добавлена возможность указывать поведение синтаксиса с помощью двойных кавычек, хотя это не обязательно.

Оператор безусловного перехода GOTO

Данный оператор реализован для поддержки программно-генерируемого кода. Но так уж повелось, что присутствие или отсутствие оператора GOTO в языке программирова-

ния всегда вызывает жаркие дебаты среди сторонников «правильного стиля» программирования. Всегда считалось признаком «хорошего тона» неиспользование данного оператора. Я не являюсь сторонником одной из этих «коалиций», хотя считаю оператор GOTO неприемлемым в серьезном программировании. Тем не менее хочу сказать, что действительно есть места, где использование оператора goto выглядит вполне логично.

Начинающие программисты, пришедшие в PHP из других языков, постоянно задаются одними и теми же вопросами. Например, такими: как выйти из многовложенного цикла? Средствами структурного программирования такой выход можно достигнуть, используя оператор break(n), но он не всегда подходит для поставленных целей. Чаще такие алгоритмы реализуются, добавляя флаги в циклы, которые проверяются при каждой итерации, что очень громоздко и непроизводительно. Процессоры не любят ветвлений, и каждая лишняя проверка съедает очень много тактов, особенно на нерегулярных переходах, которые невозможно предсказать.

Пример такого цикла может быть следующим:

```
<?php

$exit = false;
for ($i=0; $i<100; $i++)
{
    // ...

    for ($n=0; $n<100; $n++)
    {
        // ...

        for ($x=0; $x<100; $x++)
        {
            // ...

            if ( $x == 8 && 4 == $n && $i == 2 )
            {
                $exit = true;
                break;
            }

            // ...

            if ( $exit ) break;
        }

        // ...

        if ( $exit ) break;
    }

    // ...

    echo "Exit from loop: $i, $n, $x";
}

?>
```

То есть, вместо того чтобы использовать GOTO и выйти из циклов сразу, пришлось завести еще одну переменную \$exit и добавить проверки условий. Оператор GOTO в такой ситуации выглядит намного лучше. Это простой пример, но в реальности он может содержать очень много кода, и циклов может быть не три, а много больше. Кроме того, введение еще одной переменной дает возможность где-нибудь допустить ошибку, например, в ее инициализации или при проверке. Опять же, программисту придется лазить

по тексту и смотреть, зачем была нужна эта переменная.

Красивое быстрое решение для данного примера состоит в использовании горячо критикуемого GOTO, который обвиняют в неструктурности и в «идеологической неправомерности». Наш пример с использованием данного оператора будет выглядеть следующим образом:

```
<?php

for ($i=0; $i<100; $i++)
{
    // ...

    for ($n=0; $n<100; $n++)
    {
        // ...

        for ($x=0; $x<100; $x++)
        {
            // ...

            if ( $x == 8 && 4 == $n && $i == 2 )
                goto L_exit;

            // ...

        }

        // ...

    }

    // ...

L_exit:
echo "Exit from loop: $i, $n, $x";

?>
```

Как видите, данный код намного лучше предыдущего. Оператор GOTO имеет полное право на существование, потому что существуют ситуации, в которых он может значительно улучшить программный код. Но надо помнить, что таких случаев должно быть немного и программа должна проектироваться так, чтобы не возникало необходимости в данном операторе. Хотя оператор goto и может облегчить задачу, при злоупотреблении им программа превращается в спагетти, и ее становится совершенно невозможно нормально отлаживать, так как непонятно, как мы попали в этот блок кода и откуда был совершен переход сюда.

Данный оператор предназначен в первую очередь для реализации таких алгоритмов, как «конечные автоматы». В PHP у него есть также ограничение, в отличие от других языков.

Переход с помощью GOTO запрещен внутрь цикла или оператора switch.

Сокращенный тернарный оператор

Еще одним синтаксическим сахаром нас «порадовали» разработчики. Это сокращенная запись тернарного оператора «?:». Честно говоря, не совсем вижу очевидные преимущества данной записи, хотя данной фиче можно будет придумать применение. Например, такое:

```
<?php

define('IF_DEBUG', false);

// ...
```



```
function print_debug_info()
{
    print "Deug ...";
}

// ...

IF_DEBUG ?: print_debug_info();

?>
```

Правда, здесь теряется очевидность логики, так как, чтобы отработала функция `print_debug_info()`, константа `IF_DEBUG` должна принимать значение `FALSE`. Сокращенная запись по сути эквивалентна следующему выражению: «`$a ? : $b === $a ? $a : $b`». Надо сказать сразу про небольшое отличие от полной записи оператора. Следующий код показывает различия в поведении:

```
<?php
$condition = 0;

$result_1 =
$result_2 = 'True';

$result_1 = $condition ? : 'False';
$result_2 = $condition ? $result_2 : 'False';

var_dump($result_1, $result_2);

?>
```

Результат данного кода вроде бы очевиден. Обе переменные будут содержать строку `False`. Далее мы переключаем наш флаг в 1 и смотрим снова результат кода. На этот раз переменная `$result_1` будет равна 1, `$result_2` будет равна `True`. Как видите, в качестве положительного результата сокращенный оператор возвращает результат условия. Кстати, также не вижу особой полезности данного оператора, так как сокращенный вариант условий можно было писать, используя логические операторы. Мы можем написать так:

```
<?php
$a = 1;
$r2 = ,True';

$a || $r2 = ,False';

var_dump($r1, $r2);

?>
```

Правда, эта запись уже подходит под определение нетривиального синтаксиса (о котором было подробно написано в №10 за 2007 год). Тем не менее, сокращенная запись тернарного оператора может быть востребована в некоторых случаях.

Константа `__DIR__`

Частенько приходится определять константу с именем текущей директории. Как правило, это делается следующим образом:

```
<?php
define('__DIR__', dirname(__FILE__));
```

```
printf("%s \n", __DIR__);

?>
```

Теперь в этом нет необходимости, так как разработчиками PHP была добавлена константа `__DIR__`. По сути дела данная константа эквивалентна записи «`__DIR__ == dirname(__FILE__)`», однако она имеет ряд преимуществ. Одно из таких преимуществ данной константы заключается в том, что она вычисляется движком Zend Engine на стадии компиляции скрипта и соответственно потребляет меньше ресурсов, чем вышеописанный вариант. Во-вторых, она вычисляется для каждого скрипта, в котором была запрошена, в отличие от нашего варианта.

Сборщик мусора и производительность

В PHP 5.3 появился сборщик мусора, который распознает и уничтожает циклические структуры. Хотя сборщик мусора был в PHP практически с самого начала, эффективность его работы оставляла желать лучшего. Он был построен на счетчике ссылок и не распознавал циклических структур. Типичным примером такой структуры является массив, один из элементов которого является ссылкой на самого себя:

```
<?php
$a = array();
$a[0] = &$a;
unset ($a);

?>
```

Даже когда мы делали уничтожение переменной, значение счетчика ссылок не уменьшалось и не доходило до нуля. Это соответственно приводило к тому, что переменная оставалась висеть в памяти до окончания работы скрипта, занимая системные ресурсы. Сборщик мусора может запускаться как автоматически, так и явно через вызов функции `gc_collect_cycles()`. Данная функция возвращает количество уничтоженных циклических структур. Автоматический запуск сборщика мусора можно отключать функцией `gc_disable()` и включать функцией `gc_enable()` соответственно.

```
<?php
gc_disable();

$a = array();
$a[0] = &$a;
unset($a);

printf("%d \n", gc_collect_cycles());

gc_enable();

?>
```

Также выросла производительность интерпретатора по сравнению с предыдущими версиями.

Расширение системы конфигурирования

В новой версии был изменен механизм обработки файла настроек. Теперь есть возможность задавать разные настройки для разных директорий.

Например, у нас на 1-м сервере располагаются сразу девелоп-хост для разработки программного обеспечения и продакшн-хост, где выкладывается конечная версия продукта для пользователей. На девелоп-версии мы включаем вывод всех сообщений об ошибках, а на продакшн версии, естественно, отключаем вывод сообщений о каких-либо ошибках. Обычно это делается добавлением установки директивы `error_reporting` в скриптах или через вызов одноименной функции. К примеру, часто делают так: в `php.ini` директиву `error_reporting` выставляют в ноль, а в девелоп-версии скриптов в самом начале пишут «`error_reporting( E_ALL | E_STRICT )`». Не так уж это и сложно, но ведь это простой пример, а на практике может быть намного больше различных настроек, и не все они могут быть выставлены в пользовательских скриптах. Теперь появилась возможность указывать настройки для каждой директории в отдельности. Просто в `php.ini` пишем:

```
[PATH=/home/develop_host/www]
error_reporting = E_ALL | E_STRICT

[PATH=/home/production_host/www]
error_reporting = 0
```

Все. При обращении к скриптам в той или иной директории PHP будет применять соответствующие настройки. Также есть возможность задавать разные установки для разных виртуальных хостов. Мы можем предыдущий пример записать как:

```
[PATH=develop.my-super-startup.ru]
error_reporting = E_ALL | E_STRICT

[PATH= my-super-startup.ru]
error_reporting = 0
```

В итоге мы привязали настройки к виртуальным хостам, а не к директориям, что позволит нам переносить проекты из директории в директорию, но при этом не менять настройки в `php.ini`.

Более подробно о данном нововведении можно прочитать на <http://ru.php.net/ini.sections>.

Новые возможности SPL

SPL — стандартная библиотека PHP (Standard PHP Library). Представляет собой набор интерфейсов и классов, которые предназначены для решения стандартных задач и реализуют функционал для эффективного доступа к данным, интерфейсам и классам. Эта библиотека позволяет не отвлекаться на рутину по изобретению «велосипедов», так как многие алгоритмы в программировании, можно сказать, стандартизированы. SPL предоставляет программисту доступ к эффективной реализации наиболее распространенных алгоритмов и структур данных. В SPL у PHP 5.3 появился новый функционал и новые структуры данных, такие как `heap`, `stack`, `queue` и `linked list`. Много рассказывать про данные структуры не буду, так как даже начинающим программистам они должны быть известны, хотя бы из других языков программирования. Тем не менее я покажу пару примеров работы. Самый простой, с чего начнем, — это стек. Думаю, нет смысла рассказывать, что стек — это такая структура, в которой элемент, пришедший первым, будет извлечен из нее последним:

```
<?php

$Stack = new SplStack();

$Stack->push('a');
$Stack->push('b');
$Stack->push('c');

printf("%s \n", $Stack->pop());
printf("%s \n", $Stack->pop());
printf("%s \n", $Stack->pop());

?>
```

На выходе получим: c,b,a. Более сложные структуры данных, такие как `SplPriorityQueue`, позволяют формировать структуры, отсортированные по приоритету.

```
<?php

$PQueue = new SplPriorityQueue();
$PQueue->setExtractFlags( SplPriorityQueue::EXTR_DATA );

$PQueue->insert('последний элемент',1);
$PQueue->insert('первый элемент', 3);
$PQueue->insert('средний элемент', 2);

printf("Верхний элемент: %s \n", $PQueue->top());
printf("%s \n", $PQueue->extract());
printf("%s \n", $PQueue->extract());
printf("%s \n", $PQueue->extract());

?>
```

На выходе получим:

```
Приоритетный элемент: первый элемент
первый элемент
средний элемент
последний элемент
```

Более подробно о данных расширениях вы можете прочитать на <http://ru2.php.net/manual/ru/book.spl.php>.

The end...

Рассказывать о всех нововведениях можно очень долго. Но главное то, что, по заверениям разработчиков, PHP 5.3 от PHP 6 будет отличаться только поддержкой юникода. А значит, если вы научитесь работать с PHP 5.3, у вас не будет проблем с переходом на шестую версию. Да, это, конечно, произойдет не сейчас, но это произойдет. Жизнь не стоит на месте, как и прогресс. Надо всегда быть на гребне волны и смотреть в будущее. Всем известна простая истина: кто осведомлен — тот вооружен! Вооружайтесь знаниями сейчас, чтобы быть среди лидеров веб-разработки. Язык PHP «растет» и развивается, и мы, девелоперы, также развиваемся вместе с ним. Добавление таких инструментов в язык, как пространства имен и элементы функционального программирования, не говоря уже об объектно-ориентированных инструментах, позволяет наконец-то относиться к PHP как к серьезному языку программирования. Конечно, это еще пока только альфа-версия языка, но релиз не за горами. Стабильная версия PHP 5.3 ожидается уже к концу октября 2008 года. Так что в ваших интересах начать освоение новых возможностей «старого» инструмента.

1. <http://ru.php.net/manual/ru/book.spl.php>.
2. <http://wiki.php.net/doc/scratchpad/upgrade/53>.
3. <http://ru2.php.net/manual/ru/book.spl.php>.
4. <http://ru.php.net/ini.sections>.

Открываем базу данных одним щелчком мыши



Андрей Луконькин

Файловое расширение *.MD по умолчанию не обрабатывается операционной системой, но это досадное недоразумение можно исправить, добавив себе дополнительный функционал для комфортной работы.

Со временем на компьютере программиста баз «1С:Предприятие 7.7» становится слишком много, и они далеко не все зарегистрированы в списке баз для запуска (диалог «Запуск 1С:Предприятия»). Попробуем упростить себе работу, запустив «1С:Предприятие» в нужном режиме, щелкнув по файлу конфигурации 1CV7.MD.

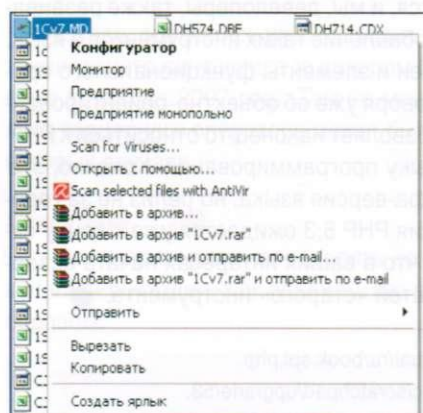
Для этого нужно зарегистрировать четыре команды контекстного меню проводника Windows для файлов

1CV7.MD: «Конфигуратор», «Монитор», «Предприятие» и «Предприятие монопольно». К тому же сделаем так, чтобы команда «Конфигуратор» была командой по умолчанию, т.е. использовалась при двойном щелчке по файлу.

В «Блокноте» (notepad.exe) или любом другом текстовом редакторе можно набрать нижеприведенный код (см. **листинг**) и сохранить его в формате TXT. Затем переименовать созданный файл, изменив расширение на REG. Таким образом, получим файл

реестра, который при запуске будет обрабатываться редактором реестра. После запуска этого файла двойным щелчком у вас должны появиться упомянутые команды в контекстном меню (см. **рисунок**).

Важно! Если у вас «1С:Предприятие» установлено в папке, отличной от «C:\Program Files\1Cv77\», или исполняемый файл называется не 1cv7s.exe, исправьте код соответствующим образом, изменив путь к исполняемому файлу и имя файла.



Так выглядит контекстное меню после описанных манипуляций с реестром

```
Windows Registry Editor Version 5.00
[HKEY_CLASSES_ROOT\*.MD]
@="mdfile"
[HKEY_CLASSES_ROOT\mdfile]
[HKEY_CLASSES_ROOT\mdfile\shell]
[HKEY_CLASSES_ROOT\mdfile\shell\open]
@="Конфигуратор"
[HKEY_CLASSES_ROOT\mdfile\shell\open\command]
@="\"C:\Program Files\1Cv77\BIN\1cv7s.exe\" config /D \"%1\..\\"
[HKEY_CLASSES_ROOT\mdfile\shell\open2]
@="Монитор"
[HKEY_CLASSES_ROOT\mdfile\shell\open2\command]
@="\"C:\Program Files\1Cv77\BIN\1cv7s.exe\" monitor /D \"%1\..\\"
[HKEY_CLASSES_ROOT\mdfile\shell\open3]
@="Предприятие"
[HKEY_CLASSES_ROOT\mdfile\shell\open3\command]
@="\"C:\Program Files\1Cv77\BIN\1cv7s.exe\" enterprise /D \"%1\..\\"
[HKEY_CLASSES_ROOT\mdfile\shell\open4]
@="Предприятие монопольно"
[HKEY_CLASSES_ROOT\mdfile\shell\open4\command]
@="\"C:\Program Files\1Cv77\BIN\1cv7s.exe\" enterprise /m /D \"%1\..\\"
```


«1С:Сценарное тестирование»

Андрей Луконькин

В сентябре вышла ознакомительная версия нового программного продукта, позволяющего повысить работоспособность и отказоустойчивость конфигураций «1С:Предприятие 8».

Новый инструмент

Инструмент состоит из двух внешних обработок. Одна обработка (ЗаписьТестов.erf) предназначена для записи теста, вторая обработка (ПрогонТестов.erf) – для прогона теста. Записанный тест можно выполнить либо в ручном, либо в автоматическом режиме. Знание программирования не требуется, что позволит использовать сценарное тестирование не только разработчикам, но также пользователям и консультантам.

Тест представляет собой набор действий, которые пользователь должен выполнить в программе. Это могут быть действия, например, по созданию новых элементов справочников, документов, заполнению данных на форме, нажатии кнопок. При автоматическом прогоне такого теста происходит имитация работы пользователя по вводу информации. Важно, что выполнение команд теста по интерактивному созданию объектов и заполнению форм отрабатывается платформой «1С:Предприятие 8» так же, как если бы эти данные вводил пользователь с клавиатуры.

В одном и том же тесте можно создавать шаги для тестирования разных хозяйственных операций. Логика теста описывается правилами отражения хозяйственных операций в программе, согласно пользовательской документации. Таким образом, инструмент можно применять для сценарного или функционального тестирования конфигураций.

Зачем это нужно?

Потребность в таком тестировании возникает, когда нужно удостовериться, что при доработке функционала конфигурации или исправлении ошибок оставшийся без изменения функционал конфигурации сохранил работоспособность. Это в большей степени востребовано в тех организациях, где идет интенсивная разработка новых релизов конфигураций, тестирование и выпуск которых имеют не меньшее значение, чем само создание нового функционала. В этом случае затраты на написание тестов и дальнейший их автоматизированный прогон будут меньше, чем при ручном тестировании каждого нового релиза конфигурации.

Как правило, такие тесты пишутся для наиболее часто используемых пользователями сценариев работы с прикладным решением, они прогоняются на каждой новой версии измененной конфигурации или платформы. Тесты можно делать более сложными или менее сложными, в зависимости от критичности ошибок в том или ином функционале прикладного

решения и в зависимости от количества времени, которое в организации готовы потратить на тестирование.

Для самых нетерпеливых приведу в качестве примера пару скриншотов работы обработок (см. рисунки).

Разработчики пока не решили, будет ли данное решение распространяться на платной или бесплатной основе, но в любом случае «1С:Сценарное тестирование» будет приобретаться и использоваться, т.к. написанная программа – это полдела, а написанная и проверенная программа – это уже завершённый проект!

К	Дата документа	Элемент	Аналитика
		Документы	
		Ввод начальных остатков	
	31.10.2009 12:00	Ввод начальных остатков 1-0000000001 от 31.10.2009	Организация1
	31.10.2009 12:00	Ввод начальных остатков 1-0000000002 от 31.10.2009	Организация1
	31.10.2009 12:00	Ввод начальных остатков 1-0000000003 от 31.10.2009	Организация1
	31.10.2009 12:00	Ввод начальных остатков 1-0000000004 от 31.10.2009	Организация1
	31.10.2009 12:00	Ввод начальных остатков 1-0000000005 от 31.10.2009	Организация1
		Продажа	
		Акт об оказании произв. услуг, Счет	
	12.01.2008 0:00	Акт об оказании производственных услуг 1-0000000001 от 12.01.2008	Организация1
	11.01.2008 12:00	Счет на оплату покупателю 1-0000000001 от 11.01.2008	Организация1
		Поступление товаров и услуг, реализация...	
	13.01.2008 12:00	Поступление товаров и услуг 1-0000000004 от 13.01.2008	Организация1
	14.01.2008 12:00	Реализация товаров и услуг 1-0000000004 от 14.01.2008	Организация1
	15.01.2008 12:00	Отчет комиссионера о продажах 1-0000000000 от 15.01.2008	Организация1
	15.01.2008 12:00	Счет-фактура выданный 1-0000000006 от 15.01.2008	Организация1
		Реализация без перехода прав собствен...	
	16.01.2008 0:00	Реализация товаров и услуг 1-0000000005 от 16.01.2008	Организация1
	17.01.2008 12:00	Реализация отгруженных товаров 1-0000000000 от 17.01.2008	Организация1
	17.01.2008 12:00	Счет-фактура выданный 1-0000000007 от 17.01.2008	Организация1
	18.01.2008 17:00	Возврат товаров от покупателя 1-0000000001 от 18.01.2008	Организация1
		Перемещение товаров, отчет о розничных...	
	20.01.2008 12:00	Перемещение товаров 1-0000000001 от 20.01.2008	Организация1
	21.01.2008 12:00	Перемещение товаров 1-0000000002 от 21.01.2008	Организация1
	22.01.2008 12:00	Отчет о розничных продажах 1-0000000001 от 22.01.2008	Организация1
	22.01.2008 12:00	Приходный кассовый ордер 1-0000000001 от 22.01.2008	Организация1
	23.01.2008 0:00	Приходный кассовый ордер 1-0000000002 от 23.01.2008	Организация1
	24.01.2008 0:00	Отчет о розничных продажах 1-0000000002 от 24.01.2008	Организация1

Настройка шага теста: Заполнить реквизиты - Заполнить реквизиты шапки

Номер шага: 15

Наименование шага: Заполнить реквизиты шапки

Реквизиты: Данные шага

Реквизит	Значение	Имя реквизита	Скрыт	Ф	0
Организация	Стройдеор	Организация			
Дата	18.04.2008 15:42:30	Дата	(TEXT), (ENTER)		
Контрагент	Поставщик основного сырья	Контрагент	(TEXT), (ENTER)		
Договор	Основной договор с поставщиком	Договор/Контрагент	(TEXT), (ENTER)		
Склад		Склад	(TEXT), (ENTER)		

Описание: Комментарий

Редактируемое описание:

Автосописание: ☐ Не включать в общее описание шага

Заполнить реквизиты:
Дата "18.04.2008 15:42:30";
Контрагент "Поставщик основного сырья";
Договор "Основной договор с поставщиком";
Проверить, что:
Организация "Стройдеор";

OK Записать Закрыть

Сквозь тернии к звездам: история компании Hewlett-Packard



Илья Александров

Их было двое. Одного звали Билл, другого – Дэвид. Они окончили университет в начале тридцатых годов прошлого века и были инженерами. Эти парни хотели заниматься технологиями. Теперь фамилии Хьюлетт (Hewlett) и Паккард (Packard) знает весь мир.

Тридцатые. Первые разработки в офисе-гараже

Билл и Дэвид окончили Стэнфорд в 1934 году, получив специальность инженера-электрика. После окончания университета они отправились на две недели на природу в Колорадо, где жили в палатках и ловили рыбу. Неформальная обстановка способствовала тому, что Хьюлетт и Паккард подружались. Билл после этого продолжил совершенствовать свои знания и навыки в Массачусетском технологическом институте (MIT), а Дэвид начал работать в компании General Electric. Они продолжили общаться и решили, что пора бы уже начать делать что-то для себя. В 1939 году они основали собственную компанию, скромно названную ими Hewlett-Packard.

Первый офис, про который так любят рассказывать в компании, являлся обычным гаражом. Начальный капитал в размере 538 долларов был пот-

рачен на покупку простенького сверлильного станка. Тогда же HP создает свой первый продукт. 200A audio – аудио-осциллятор (см. рис. 1), устройство, которое используется для тестирования работоспособности звукового оборудования. Партию из 8 моделей купил Уолт Дисней для своих звукозаписывающих студий.

Доход за первый год работы составил пять тысяч долларов.

Сороковые. Годы Второй мировой

Сороковые годы были непростыми для HP и для всей экономики в целом – Вторая мировая война вносила свои коррективы в идеи американских предпринимателей.

Как бы там ни было, компания начинает строить собственное здание для офиса и продолжает разработки. В компании отмечают, что именно в эти годы началась формироваться корпоративная этика, которая сегодня

господствует во многих американских фирмах. В HP отказываются от отдельных кабинетов, все сотрудники работают в одном большом помещении. Сейчас это называют «мульти-румом».

«Каждый сотрудник компании хочет сделать хорошее дело», – говорил Дэвид. Руководство отказывается от строгого авторитаризма, предлагая сотрудникам некоторую свободу действий в работе. Все сотрудники компании получали беспрецедентную для того времени медицинскую страховку с большими финансовыми обязательствами. Кстати, первые пожертвования на благотворительность были сделаны HP еще в 1940 году, когда доходы компании были очень скромны.

Приборы, которые производит в то время «Hewlett-Packard», нужны в основном людям научной среды и специалистам. Например, одним из главных продуктов 1942 года стал вольтметр HP 400A. В 1943 году компания начинает изготовление приборов микровол-

нового излучения, тогда эти разработки проводились для нужд военно-морского флота. HP стала признанным лидером в этой области. В 1947 году HP выпустила каталог со своей продукцией, где присутствовало 39 наименований, и Дэвид представлял проекты на нью-йоркской выставке. Несмотря на то что военные заказы от американского правительства прекратились после войны, HP сумела остаться на плаву.

К началу пятидесятых у компаний годовой оборот в 2 миллиона долларов и штат со 166 сотрудниками.

Пятидесятые. Выход на биржу и европейский рынок

В 1951 году компания выпускает продукцию для радиостанций. Радиопередатчики, измерители частоты, которые модернизировали всю радиотехническую отрасль, сделав звук и аудиооборудование более совершенным.

В 1956 году HP снова подтверждает свой статус лидера на рынке научных приборов. Новая партия осциллографов — модели 130A/150A становятся крайне популярными во многих институтах. Осциллограф — это устройство, предназначенное для визуального отображения графика электрического сигнала.

В 1957 году компания размещает свои акции на бирже, они продаются по 16 долларов за штуку. Выход на фондовый рынок — важный этап в развитии любой компании, показывающий ее состоятельность. Много изменений происходит и в самой HP.

У компании открывается огромный полноценный офис в Стэнфордском индустриальном парке, на территории располагались даже волейбольные площадки для отдыха сотрудников. Дэвид и Билл продолжают работать над корпоративной философией и этикой, Дэвид потом назовет семь главных ценностей компании: прибыль, клиенты, области, представляющие научный интерес, экономический рост, сотрудники, менеджмент и народ.

В 1958 году состоялась первая большая сделка для HP. Хьюлетт и Паккард покупают компанию F.L. Moseley, выпускавшую плоттеры. С этого момента принтеры и оргтехника становятся одним из профилирующих на-

правлений в деятельности HP. Здесь нужно сказать, что, купив чей-то бизнес, Билл и Дэвид не начинали наводить свои порядки. Приобретенная компания работала словно автономия, просто доход приносила уже новым хозяевам. Принцип «не мешать» четко соблюдался на протяжении всей истории корпорации.

Спустя год HP покупает завод в Германии и начинает массовые поставки оборудования на европейский рынок. К началу шестидесятых годовой доход составляет 48 миллионов долларов, в компании работают более двух тысяч человек.

Шестидесятые. Новые разработки

В шестидесятые годы — развитие компании, можно выделить три особые вехи. В 1964 году HP выпускает атомные часы — HP5060A. Они способны показывать точное время на протяжении трех тысячелетий с погрешностью максимум в 1 секунду. Там, где необходимо всегда знать точное время до миллисекунды, атомные часы становятся стандартом. В центрах космонавтики, у диспетчеров авиалиний и прочих.

В 1966 году HP наконец-то выпускает свой первый компьютер. Огромная машина HP 2116A изначально являлась контроллером, тестирующим на работоспособность разные электрические приборы, изготавливаемые компанией. В нем впервые инженеры компании использовали интегральные схемы. Первую модель с 4 Кб памяти купил океанографический институт и установил на одном исследовательском судне.

Наконец, в 1968 году компания выпускает настольный научный калькулятор HP 9100A. Программируемый калькулятор используется в научной среде, он умел сохранять результаты вычислений на магнитных карточках, а сами вычисления, проводимые на нем, раньше были возможны только при использовании больших компьютеров.

В этом устройстве впервые были использованы новые дисплеи, основанные на LED-индикаторах, улучшенные аналоги которых потом станут стандартом при производстве мониторов и дисплеев мобильных устройств.

HP начинает сотрудничество со странами Азии, открывает свой фи-



Рисунок 1. Осциллятор 200A audio

лиал в Японии и новый завод в США. Годовой доход в 1969 году составил 326 миллионов долларов, в компании работали 16 тысяч человек.

Семидесятые. Электронные часы и завоевание Востока

В 1972 году HP выпускает первый в мире карманный научный калькулятор. Любой инженер теперь может таскать его в кармане рубашки. HP-35 отправляет на пенсию громоздкие вычислительные приборы и логарифмическую линейку. В этом же году на рынке появляется HP-3000 — первый компьютер, ориентированный не только на работу высококвалифицированных научных работников, но и на нужды простых офисных служащих.

Компания продолжает поощрять своих сотрудников, вводит систему гибкого графика. Фактически каждый работник может сам выбрать, когда ему приходиться на работу, главное, согласовать это с начальством и отработать за день 8 часов. Забота о служащих остается приоритетом для HP. В 1977 году компания выпускает HP-01 (см. рис. 2). Наручные часы, имевшие

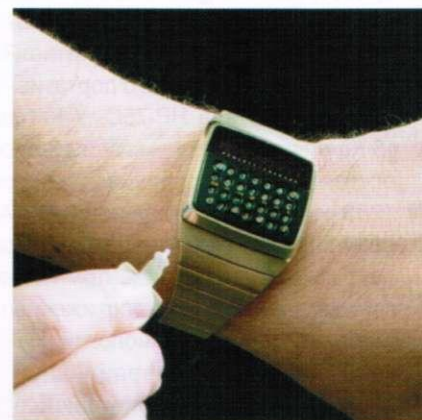


Рисунок 2. Часы HP-01



Рисунок 3. Принтер ThinkJet1984

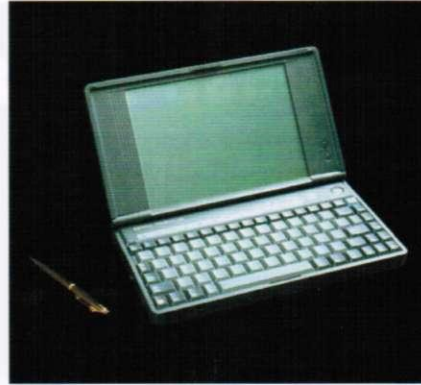


Рисунок 4. HP Omnibook 300



Рисунок 5. Компьютер HP pavilion

в своем функционале калькулятор и календарь. Просто милая, казалось бы, безделушка пользовалась бешеным успехом и казалась чем-то из мира будущего.

В это же время Билл Хьюлетт уходит в отставку с поста президента – возраст уже не позволял ему работать в полной мере. Новым президентом стал Джон Янг (John Young), начинавший в компании простым разработчиком.

Дэвид Паккард совершает несколько поездок в Китай, весьма высоко оценивая перспективы развития там бизнеса. А к началу восьмидесятых годовая выручка компании близится к двум с половиной миллиардам долларов.

Восьмидесятые. Струйные принтеры и производство компьютеров

События и выпуск новой продукции становятся такими насыщенными, что проще начать вести хронологию:

- **1980-й** – выпуск первого персонального компьютера компании, HP-85. Выпуск первого лазерного принтера.
- **1981-й** – продукция HP официально доступна для покупки в Китае.
- **1982-й** – выпуск первого портативного компьютера HP-75C. У него 16 Кб оперативной памяти и 48 Кб постоянной.
- **1983-й** – HP начинает использовать сенсорные экраны. Билл Хьюлетт награжден президентом Рейганом медалью за его достижения в области науки и технологий. Начало работы над операционной системой HP-UX, базирующейся на UNIX.

- **1984-й** – принтеры HP Thinkjet (см. рис. 3) делают струйную печать более качественной и выгодной по сравнению с матричными принтерами.
- **1985-й** – открытие завода в Китае.
- **1986-й** – HP начинает использовать в своих компьютерах архитектуру RISC.
- **1987-й** – начало массового производства 3,5-дюймовых дискет.
- **1988-й** – HP DeskJet становится одним из самых популярных принтеров в мире. Акции HP начинают продавать за пределами Америки, на Токийской бирже.
- **1989-й** – пятидесятилетие компании.

Девяностые. Портативные компьютеры

- **1990-й** – открытие филиала в Японии.
- **1991-й** – выпуск HP 95LX palmtop PC. Персональный компьютер, умещавшийся на ладони, обладал возможностью полноценного настольного компьютера.
- **1992-й** – HP представляет систему серверов HP 3000 для бизнеса.
- **1993-й** – портативный компьютер OmniBook 300 (см. рис. 4) от HP обладает рекордным временем работы от батареи. «Вы можете совершить перелет из одного конца страны в другой, а ноутбук еще будет работать».
- **1994-й** – дисплеи HP, основанные на технологии LED, являются самыми яркими и качественными в мире.
- **1995-й** – торговая марка HP Pavilion PC (см. рис. 5) становится одной из самых успешных на рынке.
- **1996-й** – умирает Дэвид Паккард.

- **1997-й** – количество сотрудников в компании достигает 122 тысяч человек.
- **1998-й** – начало производства карманных компьютеров, первая модель – Jornada PDA.
- **1999-й** – президентом компании становится Карла Фиорина (Carly Fiorina).

Наши дни

- **2000-й** – в списке успешных компаний Fortune-500 HP перемещается на 13-е место.
- **2001-й** – умирает Билл Хьюлетт. HP поглощает компанию Compaq, которая становится частью корпорации.
- **2002-й** – выпуск первого планшетного ПК Compaq Tablet PC TC1000. Научные разработки в области молекулярной электроники и нанотехнологий.
- **2004-й** – HP получает премию института электротехнических электронных инженеров (IEEE).
- **2005-й** – Персональные компьютеры Hewlett-Packard занимают первое место по продажам в Европе.

Итоги

HP сегодня – крупная технологическая корпорация с годовым доходом в 107 миллиардов долларов и мировой лидер в области продаж компьютеров. Компания продолжает развиваться и заниматься научной деятельностью, а ее первый офис-гараж навсегда останется памятником успешной истории, которую сделали два друга – Билл и Дэвид.

1. <http://www.hp.com>.
2. <http://en.wikipedia.org>.

Множественные уязвимости в Fedora

Программа: Fedora 8, 9.

Опасность: Средняя.

Описание: 1. Уязвимость существует из-за ошибки проверки границ данных в различных Red Hat Directory Server CGI-приложениях при обработке HTTP-заголовка Accept-Language. Удаленный пользователь может с помощью специально сформированного HTTP-запроса вызвать переполнение буфера и выполнить произвольный код на целевой системе с привилегиями учетной записи root.

2. Уязвимость существует из-за недостаточной обработки данных, содержащих символ «%» в Directory Server Administration Express и Directory Server Gateway (DSGW). Удаленный пользователь может с помощью специально сформированного запроса выполнить произвольный код сценария в браузере жертвы в контексте безопасности уязвимого сайта.

3. Уязвимость существует из-за недостаточной обработки входных кодированных данных, содержащих символ «%». Удаленный пользователь может с помощью специально сформированного запроса к определенным CGI-приложениям вызвать переполнение динамической памяти и выполнить произвольный код на целевой системе.

URL производителя: fedoraproject.org.

Решение: Установите исправление с сайта производителя.

Множественные уязвимости в VMware Player

Программа: VMware Player версии до 1.0.8 build 108000 и 2.0.5 build 109488.

Опасность: Высокая.

Описание: 1. Уязвимости существуют из-за неизвестных ошибок в определенных ActiveX-компонентах. Удаленный пользователь может с помощью специально сформированного веб-сайта выполнить произвольный код на целевой системе.

2. Уязвимость существует из-за неизвестной ошибки, относящейся к OpenProcess. Локальный пользователь хостовой системы может выполнить произвольный код с повышенными привилегиями на хостовой системе. Уязвимости подвержены только VMware Player 1.x for Linux.

3. Уязвимость существует из-за использования уязвимого кода библиотеки freetype. Удаленный пользователь может скомпрометировать целевую систему.

4. Уязвимость существует из-за использования уязвимой версии библиотеки cairo. Удаленный пользователь может скомпрометировать целевую систему. Уязвимости подвержены только VMware Player 2.x for Linux.

URL производителя: www.vmware.com/products/player.

Решение: Установите последнюю версию с сайта производителя.

Отказ в обслуживании в Wireshark

Программа: Wireshark, возможно, более ранние версии.

Опасность: Средняя.

Описание: 1. Различные ошибки обнаружены в epan/dissectors/packet-ncp2222.inc. Удаленный пользователь может с помощью специально сформированных NCP-пакетов вызвать зацикливание или аварийно завершить работу приложения. Уязвимости подвержены версии с 0.9.7 по 1.0.2.

2. Уязвимость существует из-за ошибки при декомпрессии zlib-сжатых данных. Удаленный пользователь может с помощью специально сформированных пакетов вызвать отказ в обслуживании приложения. Уязвимости подвержены версии с 0.10.14 по 1.0.2.

URL производителя: www.wireshark.org.

Решение: Установите последнюю версию 1.0.3 с сайта производителя.

Выполнение произвольного кода в Dns2tcp

Программа: Dns2tcp версии до 0.4.1.

Опасность: Высокая.

Описание: Уязвимость существует из-за знаковой ошибки при обработке запросов или ответов в функции dns_simple_decode() в файле common/dns.c и в функции dns_decode() в файле server/dns_decode.c. Удаленный пользователь может с помощью специально сформированного запроса или ответа вызвать повреждение памяти и вызвать отказ в обслуживании или выполнить произвольный код на целевой системе.

URL производителя: www.hsc.fr/ressources/outils/dns2tcp/index.html.en.

Решение: Установите последнюю версию 0.4.1 с сайта производителя.

Уязвимость форматной строки в HP TCP/IP Services for OpenVMS

Программа: HP TCP/IP Services for OpenVMS 5.x.

Опасность: Средняя.

Описание: Уязвимость существует из-за ошибки форматной строки в finger-клиенте. Удаленный пользователь может с помощью специально сформированных .plan-или .project-файлов, расположенных в домашней директории пользователя, выполнить произвольный код на целевой системе. Для успешной эксплуатации уязвимости злоумышленник должен обманом заставить пользователя подключиться к злонамеренному finger-серверу.

URL производителя: h71000.www7.hp.com/network/tcpip.html.

Решение: В настоящее время способов устранения уязвимости не существует.

Отравление DNS-кэша в Ingate Firewall и SiParator

Программа: Ingate Firewall версии до 4.6.4; Ingate SiParator версии до 4.6.4.

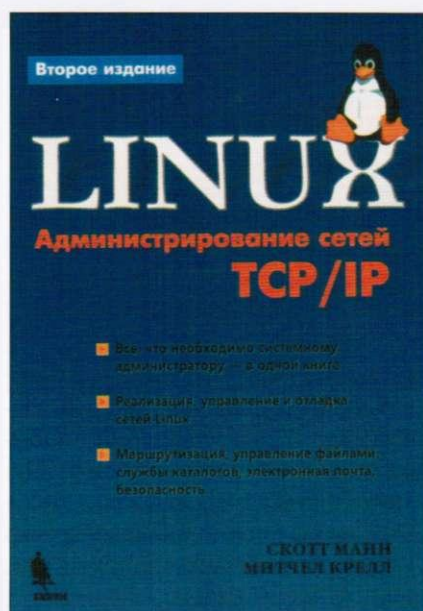
Опасность: Средняя.

Описание: Уязвимость существует из-за того, что Ingate Firewall и SiParator используют предсказуемый номер порта при включенном NAT. Удаленный пользователь может отправить кэш DNS-сервера и произвести спуфинг-атаку.

URL производителя: www.ingate.com.

Решение: Установите последнюю версию 4.6.4 с сайта производителя.

Составил Александр Антипов



Книга является обновленным переизданием оригинала, вышедшего пять лет назад. Авторы, начиная с основ, доходчиво рассказывают о реализации сетевого взаимодействия систем GNU/Linux и работе стека TCP/IP. В первую очередь издание будет интересно сетевым администраторам, строящим свою сетевую инфраструктуру на основе Linux. Действительно, не каждому системному администратору потребуются рассматриваемые в книге реализации RIP, OSPF, и уж тем более BGP.

Linux. Администрирование сетей TCP/IP

Скотт Манн, Митчел Крелл

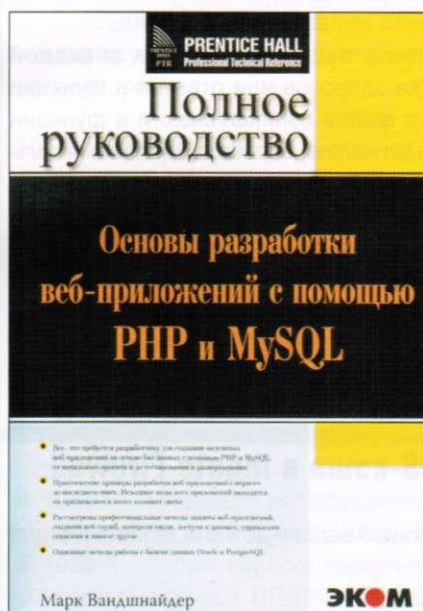
Помимо обширного теоретического материала в книге приведено множество практических примеров, которые помогают лучше понять рассматриваемые темы.

Авторы последовательно рассматривают физический и канальный уровни, протоколы ARP и RARP, работающие между канальным и сетевыми уровнями, IP версии 4 и версии 6, маршрутизацию на сетевом уровне, транспортный уровень, прикладной уровень, утилиты отладки и мониторинга (например ping, traceroute, Etherreal), кратко рассмотрены сетевые приложения (DNS, NFS, NIS, LDAP, электронная почта), дано введение в динамическое управление таблицами маршрутов, протокол OSPF и его реализация при помощи демона gated, протокол BGP, отдельная глава посвящена ipchains, что в общем-то странно для столь свежей книги, рассмотрен сетевой фильтр Netfilter, а также iproute2 и другие вопросы маршрутизации.

Несмотря на то что в книге ряд материалов устарел и не рассмотрены современные реализации некоторых протоколов, например Quagga, она все равно остается отличным пособием для сетевых администраторов, работающих с операционной системой GNU/Linux.

Издательство: «Бином-Пресс»
Год издания: 2008
Количество страниц: 672
ISBN: 978-5-9518-0230-9
Цена: ≈ 492 руб.

Книга предоставлена издательством «Бином-Пресс».



Неудивительно, что из номера в номер в обзорах появляются книги, посвященные PHP и MySQL. LAMP является одной из популярнейших связок для разработки веб-приложений, что не может не отразиться на рынке технической литературы. Автор книги – профессиональный разработчик, прошедший в компании Microsoft более пяти лет,

Основы разработки веб-приложений с помощью PHP и MySQL

Марк Вандшнайдер

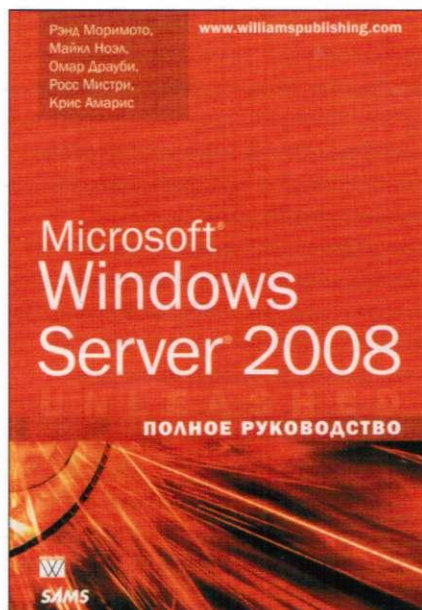
прежде чем начал работать над программным обеспечением с открытым кодом. Процесс разработки веб-приложений в книге рассматривается с системной точки зрения. Помимо написания кода затрагиваются такие темы, как интерфейс пользователя, проектирование базы данных, безопасность, тестирование и многое другое.

Книга состоит из шести частей. Первая часть посвящена основам PHP. В ней рассматриваются такие темы, как основные конструкции языка, организация и повторное использование кода, понятие об объектно-ориентированном программировании, работа с массивами, взаимодействие с сервером и интернационализация. Вторая часть знакомит читателя с основами работы с базами данных, включая работу с языком SQL. Третья часть посвящена планированию веб-приложения: реализация интерфейса пользо-

вателя, управление учетными записями, защита веб-приложений. Четвертая часть называется «Реализация веб-приложений». В ней рассмотрены вопросы отладки кода, Cookies и сеансы, аутентификация пользователей, проверка данных при помощи регулярных выражений, XML и XHTML, работа с файлами и каталогами, загрузка файлов на удаленный сервер, работа с датами и временем, веб-службы на основе XML и SOAP, использование PEAR. В пятой части приведены примеры проектов и обсуждаются дальнейшие идеи развития приложений.

Издательство: «ЭКОМ»
Год издания: 2008
Количество страниц: 828
ISBN: 978-5-9790-0054-1
Цена: ≈ 644 руб.

Книга предоставлена интернет-магазином Books.Ru.



Microsoft Windows Server 2008. Полное руководство

Рэнд Моримото и другие

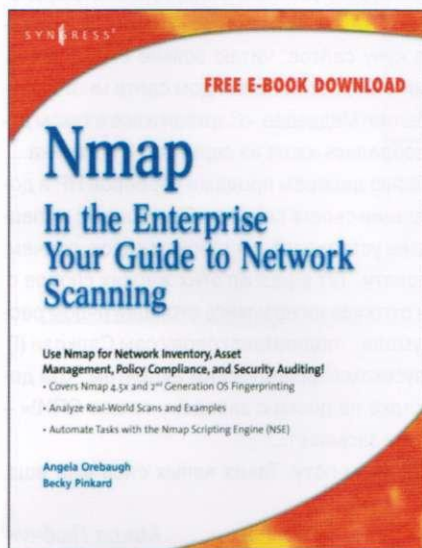
Полнотное издание объемом почти полторы тысячи страниц было бы совершенно неправильно. Издание состоит из одиннадцати частей. В первой приведен обзор Windows Server 2008, где рассказано об отличиях различных редакций друг от друга и об установке продукта. Вторая часть повествует об Active Directory в Windows Server 2008: от проектирования до внедрения. В третьей части идет речь об сетевых службах: DNS, DHCP, WINS и IIS. Также затрагиваются вопросы поддержки IPv6. Четвертая часть посвящена вопросам безопасности: безопасность на серверном уровне, на транспортном уровне, политики безопасности, сервер сетевых политик и защита сетевого доступа. В пятой части идет речь о таком вопросе, как миграция с Windows Server 2000/2003. Шестая часть затрагивает такие вопросы, как администрирование Windows Server

2008, групповые политики, практика управления и обслуживания, автоматизация задач при помощи скриптов PowerShell. Отдельная глава посвящена удаленным и мобильным технологиям. Несмотря на то что в книге речь идет именно о сервере, авторы сочли необходимым вынести в отдельную часть вопросы взаимодействия с клиентскими операционными системами. В оставшихся частях рассказано про технологии обеспечения отказоустойчивости, оптимизацию, отладку и решение проблем и про интегрированные службы приложений Windows.

- Издательство: «Вильямс»
- Год издания: 2008
- Количество страниц: 1392
- ISBN: 978-5-8459-1445-3
- Цена: ≈ 1129 руб.

Книга предоставлена издательством «Вильямс».

На книжных полках отечественных книжных магазинов появилось первое подробное руководство по Microsoft Windows 2008 Server. Первое, потому что до этого выходившие книги имели характер обзорного повествования или рассказывали лишь о новинках, появившихся во флагманском продукте Microsoft. А упрекнуть же в поверх-



Nmap in the Enterprise: Your Guide to Network Scanning

Angela Orebaugh, Becky Pinkard

с точки зрения злоумышленника и проанализировать защищенность хостов. Автор книги – признанный специалист в области безопасности, из-под его пера ранее вышли книги о таких популярных инструментах, как Wireshark (ранее Ethereal) и система предотвращения вторжений Snort.

Книга начинается с введения в сетевое сканирование в целом и программу Nmap в частности.

Из первых глав вы научитесь устанавливать Nmap в самых распространенных операционных системах: Windows, GNU/Linux, Mac OS X.

Далее подробно разбирается работа с Nmap из командной строки и GUI Zenmap, рассматриваются различные алгоритмы сканирования. В книге приведено множество примеров сканирования из реального мира, иллюстриру-

ющих работу с продуктом. Также уделено достаточно внимания и различным утилитам, расширяющим функционал Nmap, облегчающим работу с ним, в том числе NDiff, RNmap, Bilbo, Nmap-Parser.

Отдельная глава посвящена методике удаленного определения операционной системы исследуемого хоста и ее версии.

Хотя книга не очень велика по объему, но если вы хотите разобраться с таким полезным инструментом, как Nmap, то, безусловно, она заслуживает вашего внимания.

- Издательство: Syngress
- Год издания: 2008
- Количество страниц: 384
- ISBN: 978-1-5974-9241-6
- Цена: ≈ \$43,16

Вряд ли хотя бы один из системных администраторов, заботящихся о безопасности вверенных ему систем, не слышал о таком многофункциональном инструменте, как Nmap. Это программное обеспечение с открытым кодом, с успехом используемое «плохими парнями», будет нелишним в арсенале любого системного, сетевого администратора или администратора безопасности, позволяя взглянуть на сеть

Обзор книжных новинок подготовил Андрей Маркелов

Их поменяли местами

Юзеры поменялись местами, самостоятельно переподключив системные блоки. Через 5 минут одна прибегает: «Пока представляли, с рабочего стола исчезло большинство значков!!!» Еще через 10 минут второй: «В Outlook Express пропала папка с письмами!!!»

Я: «А под столом смотрели?»

Лихолет

Как пользоваться интернет-магазином

На прошлой неделе ко мне подошла сотрудница лет сорока и принесла служебную записку, подписанную генеральным, о том, что он просит предоставить ей доступ к одному ресурсу для покупки кастрюль начиная с этого момента и по момент доставки кастрюль в офис.

Через минуту я открыл доступ и в поминальник добавил напоминание о том, что надо поинтересоваться, пришли кастрюли или нет, чтобы доступ снова закрыть.

Сегодня заходит генеральный и очень деликатно интересуется, почему его указание не выполнено. Я проверяю конфиг и говорю, что доступ открыт с прошлого четверга. Предлагаю залезть с помощью gadmin на компьютер жалобщицы и убедиться в этом. Генеральный предлагает сходить к ней и разобраться на месте. Пришли... Я спрашиваю:

– В чем проблема?

– Я дважды заказала набор кастрюль, а мне не звонят.

– Покажите, как вы это делали. Какие сообщения в почте от этого магазина?

Милая дама открывает в браузере витрину магазина и тащит мышью набор кастрюль в корзину Windows, сказав в конце: «Вот и все!»

Мы с генеральным смеялись минут пять...

Потом я заказал кастрюли, позвонил в магазин, на завтра обещали доставку.

Лука М.

Как включить сеть?

Взяли новую сотрудницу. Показали рабочее место. Спрашиваю, мол, чего умеете-с. Она начала: «Я умею...» В общем, длинный список программ выдала (я аж пару раз отлучался во время ее изложения). Короче, села за комп, а я напротив как раз другой комп починал...

Через минут 30 она говорит мне: «Как сеть включается?» Я ей: «Все подключения там уже есть, просто кликайте на «Сетевое окружение» и все». Еще через полчаса слышу ее мысли якобы вслух: «Может на кнопку VIDEO щелкнуть?» Думаю, че это она тока пришла и уже страдает переизбытком свободного времени – видео надумала смотреть. Непорядок... Спустя некоторое время: «Не могли бы вы подойти?» Подхожу и вижу такую картину – сидит она за выключенным компом и тычет в кнопку VIDEO на корпусе моника! Зверее юзера я не видывал пока.

PS: под словом сеть, оказывается, понималось питание компьютера.

Дмитрий

У меня нет Интернета

Честно, надоели юзвери, которые не могут работать с Интернетом!!!

Звонит пользователь (блондинка): «У меня нет Интернета!!!»

Объясняю, как запустить IE. Говорю: «В адресной строке нужно набрать нужный вам адрес и нажать <Enter>».

Сиж, работаю дальше. Через 3 часа снова звонок от нее:

– У меня все равно не работает Интернет!!! (Некоторые сайты закрыты, и проставлена своя страничка, что данный сайт этому пользователю не разрешен.)

– Вы стерли старый адрес и вбили туда нужный адрес?

– Там написано, что Интернет не работает!!! (На мои слова ноль.)

– Вы Интернетом вообще пользовались?

– У меня никогда не было Интернета.

– Сотрите старый адрес, в эксплорере находится примерно на 3 строчке сверху, и вбейте туда, например, mail.ru, и нажмите Enter.

– Все равно не работает!!!

Мне это надоело, лезу на ее машину и вижу картину – адрес она стерла и все, ничего не вбивала. Вбиваю туда google.ru и нажимаю Enter. Та орет в трубку:

– Ура заработал!!!

Что делать? Сразу в топку?

den_neo

Сон админа

Я сейчас подбираю новый сервер для компании, ну и смотрю, анализирую, сопоставляю кучу характеристик (цена, производительность и пр.), обсмотрел кучу сайтов, читаю всякие статьи и т. д. Ну и в связи с событиями в Ю.Осетии на каждом сайте много ссылок на политические события (Медведев – Саркози и все в таком духе). В общем, в голове собралась каша из серверов + политика...

Ночью сон. Мол, я звоню дилерам продажи серверов HP и договариваюсь о конфигурации своего сервера. Спрашиваю о спецценах на сайте, меня это не устраивает, и я требую скидок, причем я делаю это лежа на кровати. Тут в разгар этих жарких споров о ценах на жесткие диски с откуда ни возьмись стоящей рядом раскладушки... чтобы вы думали... поднимает голову сам Саркози (!) и говорит на ломаном русском с французским акцентом: «Да договорюсь я о 5%-ной скидке на диски с заводом, только СПИ!» – и кладет обратно голову, и засыпает...

Я просыпаюсь в холодном поту. Таких явных снов мне еще не снилось!

Миша Любчук

Флэшка

Получил флешки. Раздал сотрудникам. Потом полдня объяснял одному дяде-полиглоту, что установить флешку на фотоаппарат я не могу, мол, флешка для компа. В результате жалоба: «Он мне вспышку на фотоаппарат отказался поставить».

hot_tabych

По материалам сайта «Сисадмин тоже человек» – <http://sysadmin.mail.ru>

Российская Федерация

- Подписной индекс: годовой – **20780**, полугодовой – **81655**
Каталог агентства «Роспечать»
- Подписной индекс: годовой – **88099**, полугодовой – **87836**
Объединенный каталог «Пресса России»
Адресный каталог «Подписка за рабочим столом»
Адресный каталог «Библиотечный каталог»
- Альтернативные подписные агентства:
Агентство «Интер-Почта» (495) 500-00-60, курьерская доставка по Москве
Агентство «Вся Пресса» (495) 787-34-47
Агентство «Курьер-Пресссервис»
Агентство «ООО Урал-Пресс» (343) 375-62-74
ЛинуксЦентр www.linuxcenter.ru
- Подписка On-line
<http://www.arzi.ru>
<http://www.gazety.ru>
<http://www.presscafe.ru>

СНГ

В странах СНГ подписка принимается в почтовых отделениях по национальным каталогам или по списку номенклатуры «АРЗИ»:

- **Азербайджан** – по объединенному каталогу российских изданий через предприятие по распространению

печати «Гасид» (370102, г. Баку, ул. Джавадхана, 21)

- **Казахстан** – по каталогу «Российская Пресса» через ОАО «Казпочта» и ЗАО «Евразия пресс»
- **Беларусь** – по каталогу изданий стран СНГ через РГО «Белпочта» (220050, г. Минск, пр-т Ф. Скорины, 10)
- **Узбекистан** – по каталогу «Davriy nashrlar» российские издания через агентство по распространению печати «Davriy nashrlar» (7000029, г. Ташкент, пл. Мустакиллик, 5/3, офис 33)
- **Армения** – по списку номенклатуры «АРЗИ» через ГЗАО «Армпечат» (375005, г. Ереван, пл. Сасунци Да-вида, д. 2) и ЗАО «Контакт-Мамул» (375002, г. Ереван, ул. Сарьяна, 22)
- **Грузия** – по списку номенклатуры «АРЗИ» через АО «Сакпресса» (380019, г. Тбилиси, ул. Хошараульская, 29) и АО «Мацне» (380060, г. Тбилиси, пр-т Гамсахурдия, 42)
- **Молдавия** – по каталогу через ГП «Пошта Молдовей» (МД-2012, г. Кишинев, бул. Штефан чел Маре, 134) по списку через ГУП «Почта Приднестровья» (MD-3300, г. Тирасполь, ул. Ленина, 17) по прайс-листу через ООО Агентство «Editil Periodice» (МД-2012, г. Кишинев, бул. Штефан чел Маре, 134)
- Подписка для **Украины**:
Киевский главпочтамт
Подписное агентство «KSS», тел./факс (044)464-0220

Ф.СП-1

Министерство связи РФ

АБОНЕМЕНТ на журнал

Системный

администратор

(индекс издания)

Количество комплектов:

на 200 год по месяцам

1	2	3	4	5	6	7	8	9	10	11	12

Куда

(почтовый индекс)

(адрес)

Кому

(фамилия, инициалы)

ДОСТАВОЧНАЯ КАРТОЧКА

ПВ место ли-тер

на журнал

(индекс издания)

Системный администратор

Стои-мость

по каталогу

руб. коп.

за доставку

руб. коп.

Количество комплектов:

на 200 год по месяцам

1	2	3	4	5	6	7	8	9	10	11	12

Куда

(почтовый индекс)

Кому

(адрес)

(фамилия, инициалы)

Подписные индексы:**20780***

+ диск с архивом статей 2008 года

81655**

без диска

по каталогу агентства «Роспечать»

88099*

+ диск с архивом статей 2008 года

87836**

без диска

по каталогу агентства «Пресса России»

* Годовой

** Полугодовой

*** Диск вкладывается в февральский номер журнала, распространяется только на территории России

УЧРЕДИТЕЛИ
Частные лица

РЕДАКЦИЯ

Генеральный директор
Владимир Положевец
Главный редактор
Алексей Коршунов
chief@samag.ru
Ответственный секретарь
Наталья Хвостова
sekretar@samag.ru
Технический редактор
Владимир Лукин

**Главный редактор
электронного приложения
«Open Source»**
Дмитрий Шурупов

Внештатные редакторы
Алексей Барабанов
Кирилл Сухов
Сергей Супрунов
Олег Щербаков
Василий Гусев

РЕКЛАМНАЯ СЛУЖБА
тел./факс: (495) 628-8253
Евгения Тарабрина (доб. 119)
reclama@samag.ru

Верстка и оформление
maker_up@samag.ru

Дизайн обложки
Дмитрий Репин

**По вопросам распространения
обращайтесь по телефону:**
Светлана Зобова
(495) 628-8253 (доб. 121)

107045, г. Москва,
Ананьевский переулок, дом 4/2, стр. 1
тел./факс: (495) 628-8253
Сайт журнала: www.samag.ru

ИЗДАТЕЛЬ
ООО «С 13»
Отпечатано типографией
ООО «Периодика»
Тираж 17000 экз.

Журнал зарегистрирован в Министерстве РФ
по делам печати, телерадиовещания и средств
массовых коммуникаций (свидетельство ПИ
№ 77-12542 от 24 апреля 2002 г.).

За содержание статьи ответственность несет
автор. Мнение редакции может не совпадать
с мнением автора. За содержание рекламных
материалов ответственность несет рекламо-
датель. Все права на опубликованные мате-
риалы защищены.

Вышел DVD-диск с архивом номеров журнала за 2007 год

Что на диске?

- Архив 12-ти номеров журнала «Системный администратор» за 2007 год.
- Архив всех выпусков электронного приложения «Open Source» за 2007 год.
- Программные продукты от партнеров журнала (триальные версии):
 - ✓ Kerio MailServer 6.5;
 - ✓ Kaspersky® Internet Security 7.0;
 - ✓ Антивирус Касперского® 7.0;
 - ✓ Система контроля доступа Zlock 2.0;
 - ✓ Прокси-сервер UserGate 4.2;
 - ✓ Handy Backup 5.8;
 - ✓ Paragon Drive Backup 8.51 Enterprise Server Edition;
 - ✓ Network Inventory 1.6.0;
 - ✓ FastReport Server 1.0.10 demo;
 - ✓ ABBYY Lingvo 12;
 - ✓ WinRAR 3.71;
 - ✓ Linux-дистрибутив Mandriva One 2008;
 - ✓ Linbox Rescue Server 20070703.
- Серверы/сети/администрирование:
 - ✓ FreeBSD 7.0-RELEASE;
 - ✓ Apache HTTP Server 2.2.8 (+ win32);
 - ✓ nginx 0.6.26, 0.5.35;
 - ✓ lighttpd 1.4.18;
 - ✓ ProFTPD 1.3.1;
 - ✓ PureFTPd 1.0.21;
 - ✓ vsftpd 2.0.6;
 - ✓ Sendmail 8.14.2;
 - ✓ Postfix 2.5.1, 2.6-20080221 (experimental);
 - ✓ Courier Mail Server 0.58.0;
 - ✓ Exim 4.69;
 - ✓ SpamAssassin 3.2.4;
 - ✓ DSPAM 3.8.0;
 - ✓ Policy Daemon 1.82;
 - ✓ SquirrelMail 1.4.13;
 - ✓ RoundCube 0.1-rc2;
 - ✓ MySQL 5.0.51a (+ win32);
 - ✓ PostgreSQL 8.3.0 (+ win32);
 - ✓ Firebird 2.0.3.12981 (+ win32);
 - ✓ SQLite 3.5.6 (+ win32);
 - ✓ Perl 5.10.0;
 - ✓ PHP: 5.2.5 (+ win32);
 - ✓ Python 2.5.2 (+ win32);
 - ✓ Ruby 1.8.6-p111, 1.9.0-1;
 - ✓ Nmap 4.53 (+ win32);
 - ✓ Ettercap 0.7.3.tar.gz (+ win32);
 - ✓ Ethereal 0.99.0 (+ win32);
 - ✓ Snort 2.8.0.2;
 - ✓ MRTG 2.16.1;
 - ✓ NuFW 2.2.9;
 - ✓ tcpdump 3.9.8;
- Пользовательское ПО:
 - ✓ NetAMS 3.4.1rc1;
 - ✓ IPStat 0.9.7.2;
 - ✓ Linux kernel 2.6.24.1;
 - ✓ BIND 9.4.2 (+ win32);
 - ✓ OpenSSL 0.9.8g;
 - ✓ OpenSSH 4.7p1;
 - ✓ Samba 3.0.28;
 - ✓ Squid Web Proxy Cache 2.6.STABLE18;
 - ✓ CUPS 1.3.6;
 - ✓ Clam AntiVirus 0.92.1 (+ Clamwin 0.92);
 - ✓ Webmin 1.400;
 - ✓ Bacula 2.2.8;
 - ✓ Amanda 2.5.2p1;
 - ✓ OpenOffice.org 2.3.1 (LinuxIntel, Win32Intel);
 - ✓ Mozilla Firefox 2.0.0.12 (Linux, win32);
 - ✓ Mozilla Thunderbird 2.0.0.12 (Linux, win32);
 - ✓ Mozilla Seamonkey (Linux, win32);
 - ✓ Mozilla Sunbird 0.7 (Linux, win32);
 - ✓ Pidgin 2.4.0 (+ win32);
 - ✓ SIM IM 0.9.4.3 (+ win32);
 - ✓ X-Chat 2.8.4 (+ 2.8.5e win32);
 - ✓ aMule 2.1.3 (+ win32);
 - ✓ GIMP 2.4.5 (+ 2.4.4 win32);
 - ✓ MPlayer 1.0rc2 (+ win32);
 - ✓ VLC media player 0.8.6e (+ win32);
 - ✓ WINE 0.9.56;
 - ✓ DOSBox 0.72 (+ win32);
 - ✓ DOSEMU 1.4.0;
 - ✓ Midnight Commander 4.6.1;
 - ✓ Vim 7.1 (+ gvim71);
 - ✓ 7-Zip 457 (+ win32);
 - ✓ GnuPG 2.0.8 (+ w32cli 1.4.8).

Как получить диск в подарок?

- Клиенты компании SecuriT, заказавшие Zserver в любой комплектации или более 50 лицензий Zlock, получают диск в подарок.

Где можно приобрести диск?

- В редакции журнала «Системный администратор» по адресу: г. Москва, Ананьевский переулок 4/2, стр. 1, в офисе 13 (метро «Сухаревская»).
- В online-магазине Linuxcenter.ru.
- В интернет-супермаркете Softkey.ru.
- Для обитателей и посетителей Савеловского рынка, диск можно купить в точке торговли печатными изданиями у главного входа.

Сколько стоит?

Цена в редакции – 150 рублей.

■ ALT Linux 4.0 Server

- Виртуализация
- Удобство настройки и сопровождения
- Поддержка аппаратных платформ x86 и x86-64



Виртуальный сервер одним нажатием кнопки

Наличие в комплекте готовых профилей для создания виртуальных серверов позволяет быстро организовать выполнение наиболее распространённых задач.

Благодаря своим возможностям ALT Linux 4.0 Server обеспечит надёжное и безопасное функционирование корпоративных сетей.

Гарантированный срок поддержки дистрибутива: 3 года

Реклама



www.altlinux.ru
sales@altlinux.ru

Так видит журнал читатель, который забыл оформить подписку:



Так видит журнал читатель, оформивший подписку:



ПОДПИШИТЕСЬ И ЧИТАЙТЕ!

Роспечать – 20780, 81655

Пресса России – 88099, 87836

Интер-почта – тел. (495) 500-00-60